WHEN DIFFUSION RECOMMENDERS STOP SAMPLING: PRESERVING GENERATIVE DIVERSITY

Anonymous authors
Paper under double-blind review

ABSTRACT

A user history can admit multiple plausible next interactions, yet conventional sequential recommenders produce a single deterministic ranking. While diffusion models offer a natural way to sample alternative continuations, we show that existing target-item diffusion recommenders often suffer from severe sampling collapse, producing nearly identical outputs across independent draws. We identify two ubiquitous design choices that promote this collapse: joint learning of the latent space and denoiser, and in-trajectory supervision via discrete losses. To address these failure modes, we introduce **DARTS** (**D**iverse and **A**ccurate **R**ecommendation **T**hrough **S**ampling), a three-stage latent diffusion framework that explicitly decouples representation learning, generative modeling, and discrete ranking by learning and freezing the latent space before diffusion training and postponing ranking supervision to the final decoding stage. We further characterize sampling diversity using complementary measures of variation before and after decoding, and use posterior-predictive aggregation to exploit this diversity at inference. Across three Amazon Reviews 2023 categories, DARTS preserves substantially greater sampling diversity while outperforming strong sequential, generative-retrieval, and diffusion baselines by 7–12% on standard ranking metrics. Aggregating multiple samples further improves ranking accuracy at a fixed slate size.

1 INTRODUCTION

Sequential recommendation aims to predict the conditional distribution of users' future interactions given their past behavior. Most commonly, these systems are trained to predict the next item given the preceding interactions. Given the diversity of user behaviors and the large size of the item space, these conditional distributions can be difficult to model accurately.

The dominant approach relies on Transformer-based recommenders, from SASRec (Kang & McAuley, 2018) and BERT4Rec (Sun et al., 2019) to their extensions to industrial-scale catalogues (Zhai et al., 2024) and semantic identifiers (Rajput et al., 2023). Despite architectural differences, they share one training paradigm: the history is encoded into a single representation, scored against the catalogue, and optimised with a catalogue cross-entropy or ranking loss. Although effective, such discriminative models may struggle to capture the diversity of user behaviour. Their output distribution can in principle be sampled to obtain varied predictions, but it does not faithfully reflect that diversity, and sampling from it degrades accuracy (Figure 4). These models are therefore used deterministically, returning the same ranking for the same history.

Diffusion models (Ho et al., 2020; Song et al., 2021) provide an alternative to discriminative models for learning complex distributions, and a growing body of work has recently incorporated them into recommender systems (Wang et al., 2023; Zhao et al., 2024; Li et al., 2024; Yang et al., 2023; Chen et al., 2025; Liu et al., 2025; Hu et al., 2025). Because generation starts from random noise, two samples conditioned on the same user history can, in principle, lead to different items. This gives diffusion-based recommenders the potential to preserve multiple plausible continuations rather than collapsing them into a single prediction (Li et al., 2024; Zolghadr et al., 2024). In practice, however, we find that this diversity largely fails to materialize. In most settings, repeatedly sampling existing

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

diffusion recommenders for the same user history yields nearly identical rankings and only exhibit a weak diversity (Section 4.3), so that the resulting sampler behaves almost deterministically.

We attribute this behavior to two design choices. First, item embeddings are learned jointly with the denoiser, allowing the latent space to reshape itself until little uncertainty remains for the denoising process to model. Second, the denoiser's intermediate outputs are supervised using catalogue-level cross-entropy or ranking losses, which encourage different denoising trajectories to converge towards the same high-scoring items. Said otherwise, relevance based recommenders naturally produce homogeneous and redundant lists. In this setting, we observe a tension between diversity and recommendation performance, as measured by accuracy metrics such as Hit Rate and NDCG: increasing diversity typically comes at the expense of recommendation accuracy, and vice versa. This is known as the diversity-accuracy dilemma Peng et al. (2024). Successful continuous generative models avoid these design choices. Latent diffusion models for images fix the representation before training the generative model (Rombach et al., 2022), while recent continuous language models such as ELF learn the mapping back to discrete tokens only at the final stage (Hu et al., 2026).

We carry these two principles over to sequential recommendation and introduce **DARTS** (**D**iverse and **A**ccurate **R**ecommendation **T**hrough **S**ampling), a generative recommendation model that can simultaneously improve prediction diversity and recommendation quality relative to the discriminative and diffusion-based baselines considered in our experiments. To the best of our knowledge, DARTS is among the first generative sequential recommenders to improve both sampling diversity and ranking performance, suggesting that generative modeling can capture multiple plausible user continuations without sacrificing recommendation quality. DARTS proceeds in three stages. We first pretrain a sequence autoencoder and freeze its latent space. We then train a denoiser in this fixed space using only a regression loss, without catalogue-level supervision. Finally, we fine-tune the decoder with a ranking objective. Sampling thus takes place entirely in the continuous latent space, allowing repeated draws to remain diverse.

Our contributions are the following.

- DARTS, a three-stage latent diffusion model for sequential recommendation that separates representation learning, generative training, and catalogue decoding (Section 3).
- Sampling diagnostics, complementary to ranking metrics, that measure the diversity of repeated draws in latent and item space. With them, we show that existing diffusion recommenders collapse to near-deterministic sampling, and we trace this collapse to joint latent training and in-trajectory catalogue supervision.
- Experiments on three categories of Amazon Reviews 2023 (Hou et al., 2026), where DARTS improves over discriminative, generative-retrieval, and diffusion baselines by 6.8–12.0% on standard ranking metrics. Aggregating several posterior samples into a single slate of fixed size gives further gains.

2 Related Work

Sequential recommendation. GRU4Rec (Hidasi et al., 2016) applied recurrent networks to interaction sequences, and later models moved to transformer encoders. SASRec (Kang & McAuley, 2018) is trained with causal next-item prediction, BERT4Rec (Sun et al., 2019) with masked item modelling, and S^3-Rec (Zhou et al., 2020) adds further self-supervised objectives. More recent work scales this recipe to industrial catalogues (Zhai et al., 2024), or retrieves items by generating their semantic identifiers autoregressively (Rajput et al., 2023; Zheng et al., 2025). Semantic-identifier models are generative too, but over discrete item codes. In contrast, DARTS samples continuous next-item representations, so the same history can produce different rankings.

Diffusion models for recommendation. Diffusion recommenders aim at generating conditional recommendation distribution. They fall into two families. DiffRec (Wang et al., 2023) and DDRM (Zhao et al., 2024) diffuse a user's interaction vector, while the second family, which includes DARTS, diffuses a representation of the target item conditioned on the history. In this second family, DiffuRec (Li et al., 2024) denoises the target embedding with a transformer, DreamRec (Yang et al., 2023) adds classifier-free guidance, and ADRec (Chen et al., 2025) warm-starts its denoiser from collaborative embeddings. Zolghadr et al. (2024) study the effect of offset noise and

cross-attention. Other methods change the training signal or the corruption process. PreferDiff (Liu et al., 2025) derives a ranking loss compatible with diffusion, and PreferGrow (Hu et al., 2025) diffuses discrete preference ratios. Most target-item methods learn the item representation together with the denoiser, and many add item-level losses during denoising. DARTS freezes a pretrained latent space, trains the denoiser by latent regression only, and applies ranking supervision only to the final decoder.

Latent and continuous generative modelling. Latent diffusion (Rombach et al., 2022) separates representation learning from generation by training and freezing an autoencoder first, then fitting the diffusion model in its latent space. In language, Diffusion-LM (Li et al., 2022) diffuses in embedding space but repeatedly projects back onto tokens. ELF (Hu et al., 2026) instead stays continuous until a single final discretisation, and shows that this improves both generation quality and efficiency. We bring both ideas, a fixed latent space and one final discretisation, to recommendation, and test whether they keep sampling stochastic when tokens are replaced by catalogue items.

3 METHOD

3.1 OVERVIEW

DARTS is composed by three architectural components: an encoder, a decoder, and a denoising diffusion model. The training of these components is realized in three separated stages that are key for a diffusion model for recommendation : the representation learning of the catalogue item in a latent space, the latent generation, and the item ranking. Each stage updates a different set of parameters of the model.

1. **Sequence autoencoder pretraining.** The encoder and decoder are trained with masked item modelling (Devlin et al., 2019; Sun et al., 2019), which defines a latent space tied to the item catalogue.
2. **Latent diffusion.** With the encoder and decoder frozen, a conditional denoiser learns to recover the latent of the next item from its noisy version, using the user history as context.
3. **Decoder finetuning.** With the encoder and denoiser frozen, the decoder is finetuned on sampled latents to rank the true next item first, which corrects the shift between the latents it was pretrained on and those generated at inference.

Figure 1 gives an overview, and Appendix A.4 summarises the parameters trained at each stage.

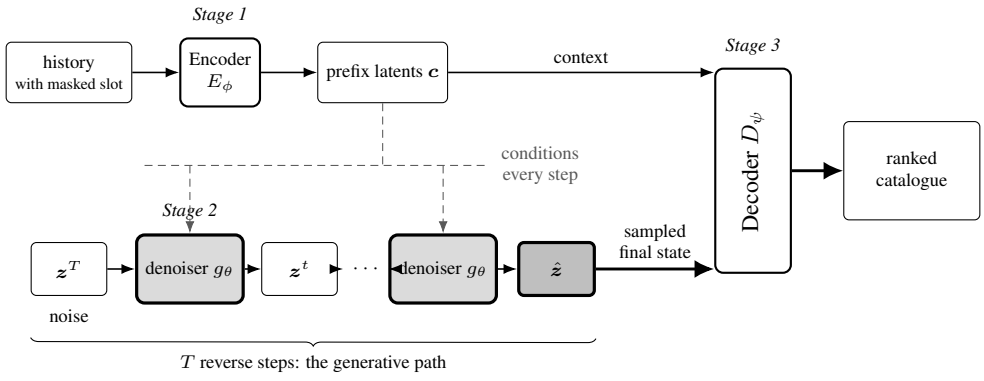


Figure 1: Model overview. The history is encoded once with its final slot masked. The prefix latents $\boldsymbol{c}$ condition every reverse step. Starting from noise, the denoiser runs T times to produce the sampled final state $\hat{\boldsymbol{z}}$. The decoder combines $\hat{\boldsymbol{z}}$ with the fixed prefix to rank the catalogue.

3.2 BACKGROUND: CONDITIONAL DENOISING DIFFUSION

To model several plausible continuations of an interaction history, we use conditional diffusion Ho et al. (2020) to learn a distribution over next-item latents. and provide below a brief background.

Forward process. Given a clean vector $\boldsymbol{z}^0$ sampled from a target distribution, diffusion defines a fixed Markov process q that gradually adds Gaussian noise over T steps (Ho et al., 2020). Let $\beta_1, \ldots, \beta_T$ be the variance schedule, $\alpha_t = 1 - \beta_t$, and $\bar{\alpha}_t = \prod_{r=1}^{t} \alpha_r$. A noisy vector at any step t can be sampled directly as

$$\boldsymbol{z}^t = \sqrt{\bar{\alpha}_t}\, \boldsymbol{z}^0 + \sqrt{1 - \bar{\alpha}_t}\, \boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{1}$$

where increasing t removes information from $\boldsymbol{z}^0$. The schedule is chosen so that $\boldsymbol{z}^T$ is approximately distributed as $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$.

Reverse process. Starting from Gaussian noise $\boldsymbol{z}^T$, generation iteratively produces $\boldsymbol{z}^{T-1}, \ldots, \boldsymbol{z}^0$. Under the $\boldsymbol{z}^0$ parameterisation, the denoiser directly estimates the clean vector as $\hat{\boldsymbol{z}}^0 = g_\theta(\boldsymbol{z}^t, t, \boldsymbol{c})$ from the current noisy vector, timestep, and condition $\boldsymbol{c}$. Substituting this estimate into the Gaussian posterior of the forward process gives the ancestral update

$$\boldsymbol{z}^{t-1} \quad = \quad \frac{\sqrt{\bar{\alpha}_{t-1}}\, \beta_t}{1 - \bar{\alpha}_t}\, \hat{\boldsymbol{z}}^0 + \frac{\sqrt{\alpha_t}\, (1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t}\, \boldsymbol{z}^t + \sqrt{\tilde{\beta}_t}\, \boldsymbol{\epsilon}, \qquad \tilde{\beta}_t = \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t}\, \beta_t, \tag{2}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ for $t > 1$ and $\boldsymbol{\epsilon} = \mathbf{0}$ for the final step. During training, a timestep is sampled according to a chosen schedule and the denoiser learns to recover $\boldsymbol{z}^0$ from $\boldsymbol{z}^t$ and $\boldsymbol{c}$.

In our context of diffusion recommendation, existing sequential diffusion recommenders Li et al. (2024); Wang et al. (2023) often use a learned item embedding as $\boldsymbol{z}^0$ and update the embedding table with the denoiser. We have a different strategy and we will clarify later the target distribution from which $\boldsymbol{z}^0$ is sampled and the condition vector.

3.3 Item representations and model backbone

Let $\mathcal{I}$ be the discrete item catalogue of size N and let $s = (i_1, \ldots, i_L) \in \mathcal{I}^L$ be a user's chronologically ordered interaction sequence of length L. For a sake of clarity, we will assume that L is fixed for all sequences. Given a binary mask $m \in \{0, 1\}^L$, its corrupted counterpart is $\tilde{s} = (\tilde{i}_1, \ldots, \tilde{i}_L) \in (\mathcal{I} \cup \{\texttt{[MASK]}\})^L$, where

$$\tilde{i}_t = \begin{cases} \texttt{[MASK]}, & m_t = 1, \\ i_t, & m_t = 0. \end{cases}$$

The recommendation task is to predict i_L from the clean prefix $i_{1:L-1}$. Note that if the sequence length is smaller than L, then the sequence is left-padded to L.

We distinguish the discrete catalogue $\mathcal{I}$ from two continuous vector spaces: the item representation space $\mathcal{V} = \mathbb{R}^{d_v}$, in which the catalogue item representations $\boldsymbol{e}_i$ live, and the latent space $\mathcal{Z} = \mathbb{R}^{d_z}$, in which the encoder representation and diffusion process resides.

Hybrid item representations. Each item $i \in \mathcal{I}$ has a representation vector $\boldsymbol{e}_i \in \mathcal{V}$ that combines semantic and collaborative information:

$$\boldsymbol{e}_i = \omega\, \boldsymbol{e}_i^{\text{sem}} + (1 - \omega)\, \boldsymbol{e}_i^{\text{id}}. \tag{3}$$

The semantic vector $\boldsymbol{e}_i^{\text{sem}}$ is a frozen `sentence-t5-base` encoding of the item's title and description (Ni et al., 2022). The identifier vector $\boldsymbol{e}_i^{\text{id}} \in \mathbb{R}^{d_v}$ is learned during Stage 1, and the mixing weight ω is a hyperparameter. We reuse the mixed item matrix as the decoder output projection, tying the input representations to the catalogue scores (Press & Wolf, 2017). Mixing the semantic and the collaborative information improved the performance, see ablation in B.

Encoder and decoder. The bidirectional transformer encoder E_ϕ maps a sequence of L item tokens to a sequence of L contextual states in $\mathcal{Z}$:

$$E_\phi(s) = \boldsymbol{z}_{1:L}, \qquad \boldsymbol{z}_j \in \mathcal{Z}.$$

Depending on the training stage, the input of the encoder may be a corrupted one $\tilde{s}$ (during autoencoder training) or a clean sequence s (when training the denoising model). The decoder D_ψ is a causal transformer that takes as input a latent sequence and outputs a sequence of items in the item representation space $\mathcal{V}$.

Hence, diffusion operates in the latent space $\mathcal{Z}$, while the decoder outputs a sequence of item representation in $\mathcal{V}$. Figure 1 shows how the components fit together. Appendix A.3 gives the block diagrams.

3.4 STAGE 1: SEQUENCE AUTOENCODER PRETRAINING

Objective. We train (E_ϕ, D_ψ) with masked item modeling. Given a training sequence $s = (i_1, \ldots, i_L)$, we sample a Bernoulli mask $m \in \{0,1\}^L$ with rate p used to form the corrupted sequence $\tilde{s} = (\tilde{i}_1, \ldots, \tilde{i}_L)$.

We denote as $\boldsymbol{h}_{1:L} = [\boldsymbol{h}_1, \cdots, \boldsymbol{h}_L]$, with $\boldsymbol{h}_j \in \mathcal{V}$, the outputs of the autoencoder $D_\psi(E_\phi(\tilde{s}))$ and we train the couple to recover the original items at the masked positions. Given

$$p_{\phi,\psi}(i \mid \tilde{s}, j) = \frac{\exp\big(\boldsymbol{h}_j^\top \boldsymbol{e}_i\big)}{\sum_{i' \in \mathcal{I}} \exp\big(\boldsymbol{h}_j^\top \boldsymbol{e}_{i'}\big)}.$$

$p_{\phi,\psi}$ being the catalogue-wide softmax distribution induced at position j by the encoder–decoder pair and the item embeddings, we consider the following loss for the autoencoder training

$$\mathcal{L}_{\text{MLM}}(\phi, \psi) \quad = \quad -\,\mathbb{E}_{s,m}\left[\frac{1}{|\mathcal{M}|}\sum_{j \in \mathcal{M}} \log p_{\phi,\psi}(i_j \mid \tilde{s}, j)\right]. \tag{4}$$

Choice of masking ratio. We use a masking ratio of 0.4 and always mask the final item to match the next-item prediction objective of sequential recommendation.

After Stage$\tilde{1}$, the encoder and item representations remain frozen during Stages$\tilde{2}$ and$\tilde{3}$. The decoder is kept frozen during Stage$\tilde{2}$ and fine-tuned during Stage$\tilde{3}$.

3.5 STAGE 2: LATENT DIFFUSION

In this stage, we train the diffusion denoiser to model the distribution of the next-item latent conditioned on the observed interaction history, while keeping the pretrained latent space fixed.

Condition and target. Diffusion training uses two encoder passes with different inputs. First, we mask the final item:

$$s^{\text{mask}} = (i_1, \ldots, i_{L-1}, \texttt{[MASK]}).$$

and encode this sequence to obtain the conditioning prefix of the diffusion denoiser

$$\boldsymbol{c} = \big[E_\phi(s^{\text{mask}})\big]_{1:L-1}. \tag{5}$$

The target distribution must relate to the distribution of final item i_L. To define the clean diffusion target $\boldsymbol{z}0$, we pass the uncorrupted sequence s through E_ϕ and extract the representation at the final slot:

$$\boldsymbol{z}^0 = \mathrm{sg}\big([E_\phi(s)]_L\big), \tag{6}$$

where sg denotes the stop-gradient operator. Consequently, the condition $\boldsymbol{c}$ uses strictly the information available at inference time, while the target $\boldsymbol{z}^0$ reflects the contextualized representation of the target item given the full sequence.

This key formulation differs from existing target-item diffusion approaches (Li et al., 2024; Yang et al., 2023; Chen et al., 2025), which diffuse isolated, static item embeddings $e(i_L)$. By contrast, our target $\boldsymbol{z}^0$ is generated by the sequence encoder E_ϕ, capturing rich context-dependent item semantics. Furthermore, because E_ϕ is frozen, both the conditioning space and the target space remain fixed throughout diffusion training, preventing representation drift.

Denoiser's Architecture. The diffusion denoiser g_θ is a stack of transformer blocks. At each block, the noisy target latent $\boldsymbol{z}^t$ acts as the cross-attention query and the prefix states $\boldsymbol{c}$ provide keys and values. The timestep modulates the block through adaptive layer normalisation (Peebles & Xie, 2023). Padding positions in the prefix are excluded from cross-attention.

Objective. For a sampled timestep t and noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, we construct

$$\boldsymbol{z}^t = \sqrt{\bar{\alpha}_t}\,\boldsymbol{z}^0 + \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}.$$

The denoiser is trained with latent root mean squared error and predicts the clean target directly.

$$\mathcal{L}_{\text{diff}}(\theta) = \sqrt{\mathbb{E}_{s,t,\boldsymbol{\epsilon}}\left[\frac{1}{d_z}\left\|g_\theta(\boldsymbol{z}^t, t, \boldsymbol{c}) - \boldsymbol{z}^0\right\|_2^2\right]}. \tag{7}$$

We want to emphasize that contrarily existting approaches, no catalogue-related loss is applied in this stage.

3.6 Stage 3: Decoder Fine-Tuning

In Stage 1, the encoder and decoder are trained jointly to reconstruct the full item sequence. At inference, however, the decoder receives the conditioning sequence $\boldsymbol{c}$ of length $L-1$ and the generated target latent $\hat{\boldsymbol{z}}_0$ at the final slot. This creates a distribution shift: slot L contains a masked context representation during pretraining but a latent sample from reverse diffusion at inference. Stage 3 closes this gap by finetuning the decoder to rank the target item above candidates using its inference inputs. The encoder E_ϕ, denoiser g_θ, and item embeddings remain frozen.

For a training sequence s, we extract the conditioning $\boldsymbol{c}$ via Equation 5, sample $\boldsymbol{z}_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, and execute the full reverse diffusion process to sample a target latent $\hat{\boldsymbol{z}}_0 \sim p_\theta(\boldsymbol{z}_0 \mid \boldsymbol{c})$. We concatenate $\boldsymbol{c}$ and $\hat{\boldsymbol{z}}_0$ to form the L-length input sequence for the decoder D_ψ. The output of interest is the logit vector $\boldsymbol{l} \in \mathbb{R}^N$ over the item catalogue at the final position.

Let $y = i_L$ be the ground-truth target item, and let $\mathcal{H}_n(\boldsymbol{l}, y)$ denote the set of n highest-scoring negative items (i.e., self-mined hard negatives). We fine-tune D_ψ using a temperature-scaled Bayesian Personalized Ranking (BPR) loss (Rendle et al., 2009):

$$\mathcal{L}_{\text{dec}}(\psi) = -\frac{1}{n}\sum_{j \in \mathcal{H}_n(\boldsymbol{l}, y)} \log \sigma\left(\frac{l_y - l_j}{\tau}\right), \tag{8}$$

where $\tau > 0$ is a temperature hyperparameter and $\sigma(\cdot)$ is the sigmoid function. Optimizing $\mathcal{L}_{\text{dec}}$ adapts the decoder to map continuous sampled latents directly to discrete catalogue items, resolving the inference distribution shift without changing the stochastic latent distribution learned in Stage 2. Inference follows the same sampling and decoding path (Appendix A.5).

4 Experiments

In this section, we empirically evaluate DARTS against competitive baselines. We further introduce diagnostic measurements to analyze the stochastic properties of the diffusion process and quantify the diversity of its predictions. We also conduct controlled ablations to isolate how common design choices in prior work induce sampling collapse.

4.1 Datasets

We evaluate DARTS on three categories of the Amazon 2023 review corpus (Hou et al., 2026): Musical Instruments (Instrument), Industrial and Scientific (Scientific), and Video Games (Game). Following prior work (Rajput et al., 2023; Zhou et al., 2020; Zheng et al., 2025), we retain users and items with at least five interactions, order each history chronologically, and keep its 20 most recent items. Appendix Table 3 reports the resulting statistics.

4.2 Overall Performance - Accuracy

Table 1 compares DARTS against three families of baselines: sequential recommenders (Hidasi et al., 2016; Sun et al., 2019; Kang & McAuley, 2018; Zhou et al., 2022; Zhai et al., 2024; Zhang et al., 2019; Zhou et al., 2020), generative retrieval with semantic identifiers (Rajput et al., 2023; Wang et al., 2024; Hou et al., 2025), and models we train ourselves with the same hybrid item representations as DARTS (*+T5*): SASRec and the diffusion recommenders DreamRec (Yang et al., 2023), DiffuRec (Li et al., 2024), and ADRec (Chen et al., 2025). Results for the first two families are taken from Zheng et al. (2025), except ActionPiece from Zhong et al. (2025); all follow the protocol above. We report Recall@K and NDCG@K for $K \in \{5, 10\}$ under leave-one-out evaluation; Appendix A details baselines, implementation, and hyperparameters.

Table 1: Recommendation performance on three Amazon Reviews 2023 categories. R@K/N@K are Recall/NDCG@K; **best** and second best are marked, and *Improv.* is our gain over the best baseline per column.

	Instrument				Scientific				Game			
Method	R@5	R@10	N@5	N@10	R@5	R@10	N@5	N@10	R@5	R@10	N@5	N@10
Sequential recommenders												
GRU4Rec	0.0324	0.0501	0.0209	0.0266	0.0202	0.0338	0.0129	0.0173	0.0499	0.0799	0.0320	0.0416
BERT4Rec	0.0307	0.0485	0.0195	0.0252	0.0186	0.0296	0.0119	0.0155	0.0460	0.0735	0.0298	0.0386
SASRec	0.0333	0.0523	0.0213	0.0274	0.0259	0.0412	0.0150	0.0199	0.0535	0.0847	0.0331	0.0438
FMLP-Rec	0.0339	0.0536	0.0218	0.0282	0.0269	0.0422	0.0155	0.0204	0.0528	0.0857	0.0338	0.0444
HSTU	0.0343	0.0577	0.0191	0.0271	0.0271	0.0429	0.0147	0.0198	0.0578	0.0903	0.0334	0.0442
FDSA	0.0347	0.0545	0.0230	0.0293	0.0262	0.0421	0.0169	0.0213	0.0544	0.0852	0.0361	0.0448
S^3-Rec	0.0317	0.0496	0.0199	0.0257	0.0263	0.0418	0.0171	0.0219	0.0485	0.0769	0.0315	0.0406
Generative retrieval with semantic identifiers												
TIGER	0.0370	0.0564	0.0244	0.0306	0.0264	0.0422	0.0175	0.0226	0.0559	0.0868	0.0366	0.0467
LETTER	0.0372	0.0580	0.0246	0.0313	0.0279	0.0435	0.0182	0.0232	0.0563	0.0877	0.0372	0.0473
ActionPiece	0.0383	0.0615	0.0243	0.0318	0.0284	0.0452	0.0182	0.0236	0.0591	0.0927	0.0382	0.0490
SASRec + T5	0.0376	0.0595	0.0243	0.0311	0.0277	0.0452	0.0177	0.0232	0.0534	0.0853	0.0344	0.0447
DreamRec + T5	0.0146	0.0269	0.0088	0.0124	0.0112	0.0196	0.0071	0.0098	0.0146	0.0249	0.0086	0.0122
DiffuRec + T5	0.0387	0.0596	0.0248	0.0315	0.0302	0.0468	0.0190	0.0243	0.0591	0.0944	0.0368	0.0482
ADRec + T5	0.0314	0.0498	0.0201	0.0260	0.0236	0.0374	0.0150	0.0195	0.0549	0.0869	0.0357	0.0459
DARTS	**0.0423**	**0.0663**	**0.0271**	**0.0348**	**0.0329**	**0.0524**	**0.0207**	**0.0270**	**0.0653**	**0.1045**	**0.0408**	**0.0534**
Improv.	+9.3%	+7.8%	+9.3%	+9.4%	+8.9%	+12.0%	+8.9%	+11.1%	+10.5%	+10.7%	+6.8%	+9.0%

In terms of these accuracy measures, DARTS performs best on all twelve combinations of datasets and metrics, improving over the strongest baseline in each column by between 6.8% and 12.0%.

Disentangling item representations from the generative formulation. SASRec + T5 shares our item representations and codebase, so this comparison isolates the generative process. Our NDCG@10 gains are 11.9% on INSTRUMENT, 16.4% on SCIENTIFIC, and 19.5% on GAME w.r.t. SASRec + T5. The diffusion baselines share the same representations yet vary widely, so our gains come from the staged design rather than from item features. Appendix B isolates the representation type. Semantic features lift our NDCG@10 by 14.4%, but our identifier-only variant still leads every identifier-only competitor by 18.6%. DiffuRec, the closest baseline since it also diffuses target representations, learns its latent space jointly with the denoiser and applies intermediate catalogue cross-entropy, two design choices we further evaluate next.

4.3 ASSESSING THE DIVERSITY-ACCURACY COOMPROMISE

Single-draw ranking metrics cannot determine whether diffusion produces meaningful stochastic variation. For each test context $\boldsymbol{c}$, we therefore draw $K = 10$ independent latents $\hat{\boldsymbol{z}}^0$ from $p_\theta(\cdot \mid \boldsymbol{c})$, decode each into a ranked list, and compare the draws.

Diagnostics. We introduce three metrics that help us evaluate the generative capability of generative recommender models in terms of diversity of the predictions.

The first one is a mean pairwise Jaccard dissimilarity measure. Assuming that $\mathcal{S}_{\boldsymbol{c}}^{(k)}$ denotes the top-10 set from draw k, we have the measure

$$J = \mathbb{E}_{\boldsymbol{c}}\left[\binom{K}{2}^{-1} \sum_{k<l}\left(1 - \frac{|\mathcal{S}_{\boldsymbol{c}}^{(k)} \cap \mathcal{S}_{\boldsymbol{c}}^{(l)}|}{|\mathcal{S}_{\boldsymbol{c}}^{(k)} \cup \mathcal{S}_{\boldsymbol{c}}^{(l)}|}\right)\right].$$

J is zero for identical slates and grows as resampling yields distinct recommendations. Because J is computed after decoding, it reflects the combined behavior of the sampler and the decoder. The next two metrics operate in latent space and isolate the stochasticity of the denoiser.

Repeated samples may fail to be generative in two ways: they may exhibit negligible variation or explore only a few latent directions. We measure these complementary forms of collapse using vari-

ance decomposition and effective dimensionality. For K draws $\{\hat{\boldsymbol{z}}_{\boldsymbol{c}}^{(k)}\}_{k=1}^{K}$, let $\bar{\boldsymbol{z}}_{\boldsymbol{c}} = K^{-1}\sum_k \hat{\boldsymbol{z}}_{\boldsymbol{c}}^{(k)}$ and $\bar{\boldsymbol{z}} = \mathbb{E}_{\boldsymbol{c}}[\bar{\boldsymbol{z}}_{\boldsymbol{c}}]$. Define $V_{\mathrm{w}} = \mathbb{E}_{\boldsymbol{c}}[K^{-1}\sum_k \|\hat{\boldsymbol{z}}_{\boldsymbol{c}}^{(k)} - \bar{\boldsymbol{z}}_{\boldsymbol{c}}\|_2^2]$ and $V_{\mathrm{b}} = \mathbb{E}_{\boldsymbol{c}}[\|\bar{\boldsymbol{z}}_{\boldsymbol{c}} - \bar{\boldsymbol{z}}\|_2^2]$. By the within–between variance decomposition, the *generative variance ratio*

$$R = \frac{V_{\mathrm{w}}}{V_{\mathrm{w}} + V_{\mathrm{b}}} \tag{9}$$

is the fraction of latent variance due to resampling rather than differences between histories. $R = 0$ indicates a deterministic point predictor.

As R quantifies the magnitude of sampling variation, but not how many directions it spans, we therefore define the *participation ratio* of the within-context sample covariance matrix $\boldsymbol{C}_{\boldsymbol{c}}$:

$$\mathrm{PR} = \mathbb{E}_{\boldsymbol{c}}\left[\frac{(\mathrm{tr}\,\boldsymbol{C}_{\boldsymbol{c}})^2}{\mathrm{tr}\,(\boldsymbol{C}_{\boldsymbol{c}}^2)}\right], \tag{10}$$

where $\boldsymbol{C}_{\boldsymbol{c}} = \frac{1}{K-1}\sum_{k=1}^{K}(\hat{\boldsymbol{z}}_{\boldsymbol{c}}^{(k)} - \bar{\boldsymbol{z}}_{\boldsymbol{c}})(\hat{\boldsymbol{z}}_{\boldsymbol{c}}^{(k)} - \bar{\boldsymbol{z}}_{\boldsymbol{c}})^\top$. The participation ratio estimates the effective dimensionality of the sampled subspace: approaching 0 for a completely collapsed point, 1 when variance is restricted to a single axis, and up to $\min(K-1, d)$ for isotropic exploration across available dimensions.

Results. Neither R nor PR measures recommendation quality, so we interpret them alongside NDCG and the decoded-set diagnostic J. Table 2 reports the three diagnostics on SCIENTIFIC. All three baselines produce nearly fixed slates, but their latent diagnostics reveal different failure modes. ADRec collapses in both latent and item space. DiffuRec explores several directions ($\mathrm{PR} = 5.0$), but at negligible scale relative to variation across histories ($R = 4.8 \times 10^{-5}$), so 98% of its top-10 is preserved between draws. DreamRec has a larger variance ratio ($R = 0.08$) but an almost zero participation ratio (0.005), indicating little latent exploration. Its low NDCG further shows why these diagnostics cannot be read as quality measures.

Table 2: Sampling diagnostics on SCIENTIFIC from $K = 10$ draws per context: Jaccard dissimilarity J, generative variance ratio R, and participation ratio PR. NDCG@5 (single draw) is repeated to show the diagnostics are not a proxy for ranking quality.

Model	$J \uparrow$	$R \uparrow$	PR $\uparrow$	NDCG@5
DreamRec + T5	0.020	8.0×10^{-2}	0.005	0.0071
DiffuRec + T5	0.019	4.8×10^{-5}	5.0	0.0190
ADRec + T5	3.0×10^{-4}	9.0×10^{-9}	2.6×10^{-5}	0.0150
DARTS	**0.30**	**0.30**	4.7	**0.0207**

DARTS combines variation ($R = 0.30$) with multi-directional exploration ($\mathrm{PR} = 4.7$), and this diversity survives decoding: $J = 0.30$, i.e. a pairwise Jaccard similarity of 0.70 between top-10 sets. It thus achieves strong diversity while outperforming all baselines in Table 1.

4.4 ABLATIONS

Catalogue cross-entropy supervision should not be used inside the diffusion process. We retrain Stage 2 from one Stage-1 checkpoint with $(1-\alpha)\mathcal{L}_{\text{diff}} + \alpha\mathcal{L}_{\text{CE}}$, where $\mathcal{L}_{\text{CE}}$ is defined in Equation 11 and compute a loss at the decoder level. The setting $\alpha = 0$ is our latent-only objective, while $\alpha = 1$ uses cross-entropy alone. Results and detailed analyses are in Table 7 and Appendix C. We note that even $\alpha = 0.1$ reduces R by more than half and lowers PR from 4.7 to 1.5. Both metrics continue to decrease as α increases, reaching 0.005 and 0.65, respectively, at $\alpha = 1$. In addition, the use of cross-entropy loss does not improve ranking either: NDCG@5 is highest at $\alpha = 0$ and decreases monotonically to 0.0089 at $\alpha = 1$, a drop of 57%. Discrete supervision therefore suppresses generative properties without improving ranking accuracy. This finding is consistent with the continuous-space argument of Hu et al. (2026).

Freezing the item representations. We next test whether the item representations should remain fixed during Stage 2 (Details in Appendix D and Table 8). Under latent regression loss, updating them jointly with the denoiser reduces R from 0.300 to 0.024, lowers PR from 4.70 to 1.30, and decreases NDCG@5 by 42%. Thus, keeping fix the item representations and excluding catalogue cross-entropy loss are necessary to obtain the strong sampling behaviour and ranking accuracy.

Generative behaviour converts into accuracy. Table 2 shows that DARTS is generative and Table 1 that a single draw is accurate; we now show that aggregating multiple draws turns this diversity into further accuracy gains. DARTS allows us to define the posterior-predictive mixture $p(i|\boldsymbol{c}) = \mathbb{E}_{\boldsymbol{z}\sim p_\theta}[p_\psi(i|\boldsymbol{z}, \boldsymbol{c})]$, where $p_\psi(i|\boldsymbol{z}, \boldsymbol{c})$ is the softmax over item scores. We estimate $p(i|\boldsymbol{c})$ by averaging p_ψ over M draws and rank items by this average at a fixed top-k. Note that a model that returns the same item distribution on every pass cannot benefit from repeated evaluation because identical distributions aggregate to themselves. Evaluating this aggregation on SCIENTIFIC (see Figure 2 and the full results in Appendix G Figure 6), DARTS improves on all four metrics from $M = 1$ to $M = 4$, by 3.2% on both NDCG@5 and NDCG@10, while DiffuRec moves by at most 0.6% and is effectively flat beyond $M = 2$. Thus, additional draws improve our ranking at fixed slate size without retraining, whereas DiffuRec's nearly coincident draws provide no benefit.

Each draw repeats the reverse chain, so this gain is not free: throughput in predictions per second falls by $3.3\times$ between $M = 1$ and $M = 4$. Appendix G reports end-to-end inference cost and shows that the $M = 1$ model is nonetheless $3.6\times$ faster than DiffuRec with better ranking accuracy, because our denoiser conditions on a single latent token instead of re-encoding the item sequence at every step. Appendix F contrasts this with pooling the top-n lists, which does not improve accuracy once the larger slate is accounted for. Finally, randomness and repeated inference alone do not explain these gains. Perturbing SASRec's catalogue distribution with Gumbel noise sharply reduces ranking accuracy, and aggregating up to $M = 15$ draws only moves the result back toward, and never above, the deterministic ranking (Appendix E and Figure 4).

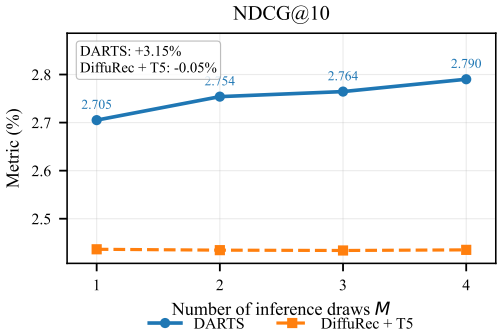


Figure 2: Posterior-predictive aggregation on SCIENTIFIC. Ranking by the arithmetic mean of M independent draws improves our NDCG@10, whereas DiffuRec remains nearly flat. Appendix G reports all four metrics.

5 DISCUSSION

Existing diffusion recommenders are generative by construction, yet their repeated samples yield nearly identical recommendation lists. We traced this collapse to two design choices, learning the representation jointly with the denoiser and supervising the reverse process with catalogue losses, and showed by ablation that each one sharply reduces latent variation and degrades ranking. Removing both keeps stochastic variation through decoding while the model leads every comparison in Table 1, and averaging several draws further improves all four ranking metrics (Figure 6), a gain that injecting noise into a discriminative model does not provide (Appendix E).

Limitations and future work. Leave-one-out evaluation observes a single next interaction, so ranking metrics cannot tell whether alternative samples are plausible futures. Multi-future settings would allow calibration, controllable diversity, and coverage to be measured directly. Structured outputs such as semantic identifiers are a natural next target as well.

AI USE STATEMENT

We used generative AI tools to polish the manuscript, primarily to correct English grammar and improve the clarity of the writing and to assist with portions of the code.

REFERENCES

Jialei Chen, Yuanbo Xu, and Yiheng Jiang. Unlocking the power of diffusion models in sequential recommendation: A simple and effective approach. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2025. doi: 10.1145/3711896.3737172. URL https://arxiv.org/abs/2505.19544.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 4171–4186, 2019. doi: 10.18653/v1/N19-1423.

Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. In *4th International Conference on Learning Representations*, 2016. URL https://arxiv.org/abs/1511.06939.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems 33*, 2020. URL https://arxiv.org/abs/2006.11239.

Yupeng Hou, Jianmo Ni, Zhankui He, Noveen Sachdeva, Wang-Cheng Kang, Ed H. Chi, Julian McAuley, and Derek Zhiyuan Cheng. ActionPiece: Contextually tokenizing action sequences for generative recommendation. In *Forty-second International Conference on Machine Learning*, 2025. URL https://arxiv.org/abs/2502.13581.

Yupeng Hou, Jiacheng Li, Xiangjun Fu, Zhankui He, An Yan, Xiusi Chen, and Julian McAuley. Bridging language and items for retrieval and recommendation: Benchmarking LLMs as semantic encoders. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3251–3265. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.147.

Guoqing Hu, An Zhang, Shuchang Liu, Wenyu Mao, Jiancan Wu, Xun Yang, Xiang Li, Lantao Hu, Han Li, Kun Gai, and Xiang Wang. Fading to grow: Growing preference ratios via preference fading discrete diffusion for recommendation. In *Advances in Neural Information Processing Systems 38*, 2025. URL https://arxiv.org/abs/2509.26063.

Keya Hu, Linlu Qiu, Yiyang Lu, Hanhong Zhao, Tianhong Li, Yoon Kim, Jacob Andreas, and Kaiming He. ELF: Embedded language flows. *CoRR*, abs/2605.10938, 2026. URL https://arxiv.org/abs/2605.10938.

Wang-Cheng Kang and Julian J. McAuley. Self-attentive sequential recommendation. In *IEEE International Conference on Data Mining*, pp. 197–206. IEEE Computer Society, 2018. doi: 10.1109/ICDM.2018.00035.

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B. Hashimoto. Diffusion-LM improves controllable text generation. In *Advances in Neural Information Processing Systems 35*, 2022. URL https://arxiv.org/abs/2205.14217.

Zihao Li, Aixin Sun, and Chenliang Li. DiffuRec: A diffusion model for sequential recommendation. *ACM Transactions on Information Systems*, 2024. URL https://arxiv.org/abs/2304.00686.

Shuo Liu, An Zhang, Guoqing Hu, Hong Qian, and Tat-Seng Chua. Preference diffusion for recommendation. In *13th International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.13117.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. Sentence-T5: Scalable sentence encoders from pre-trained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 1864–1874, 2022. doi: 10.18653/v1/2022.findings-acl.146. URL https://aclanthology.org/2022.findings-acl.146/.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *IEEE/CVF International Conference on Computer Vision*, pp. 4172–4182, 2023. doi: 10.1109/ICCV51070.2023.00387.

Kenneth L Peng, Manish Raghavan, Emma Pierson, Jon Kleinberg, and Nikhil Garg. Reconciling the accuracy-diversity trade-off in recommendations. In *The Web Conference 2024*, 2024. URL https://openreview.net/forum?id=rSHR9YKBVy.

Ofir Press and Lior Wolf. Using the output embedding to improve language models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 157–163, 2017. URL https://arxiv.org/abs/1608.05859.

Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan Hulikal Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Q. Tran, Jonah Samost, Maciej Kula, Ed H. Chi, and Mahesh Sathiamoorthy. Recommender systems with generative retrieval. In *Advances in Neural Information Processing Systems 36*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/20dcab0f14046a5c6b02b61da9f13229-Abstract-Conference.html.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, pp. 452–461, 2009. URL https://arxiv.org/abs/1205.2618.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10674–10685, 2022. doi: 10.1109/CVPR52688.2022.01042.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *9th International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2011.13456.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. doi: 10.1016/j.neucom.2023.127063.

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 28th ACM International Conference on Information and Knowledge Management*, pp. 1441–1450. ACM, 2019. doi: 10.1145/3357384.3357895.

Wenjie Wang, Yiyan Xu, Fuli Feng, Xinyu Lin, Xiangnan He, and Tat-Seng Chua. Diffusion recommender model. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 832–841. ACM, 2023. doi: 10.1145/3539618.3591663. URL https://arxiv.org/abs/2304.04971.

Wenjie Wang, Honghui Bao, Xinyu Lin, Jizhi Zhang, Yongqi Li, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. Learnable item tokenization for generative recommendation. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pp. 2400–2409. ACM, 2024. doi: 10.1145/3627673.3679569.

Zhengyi Yang, Jiancan Wu, Zhicai Wang, Yancheng Yuan, Xiang Wang, and Xiangnan He. Generate what you prefer: Reshaping sequential recommendation via guided diffusion. In *Advances in Neural Information Processing Systems 36*, 2023. URL https://arxiv.org/abs/2310.20453.

Jiaqi Zhai, Lucy Liao, Xing Liu, Yueming Wang, Rui Li, Xuan Cao, Leon Gao, Zhaojie Gong, Fangda Gu, Jiayuan He, Yinghai Lu, and Yu Shi. Actions speak louder than words: Trillion-parameter sequential transducers for generative recommendations. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=xye7iNsgXn.

Tingting Zhang, Pengpeng Zhao, Yanchi Liu, Victor S. Sheng, Jiajie Xu, Deqing Wang, Guanfeng Liu, and Xiaofang Zhou. Feature-level deeper self-attention network for sequential recommendation. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pp. 4320–4326, 2019. doi: 10.24963/ijcai.2019/600.

Jujia Zhao, Wenjie Wang, Yiyan Xu, Teng Sun, Fuli Feng, and Tat-Seng Chua. Denoising diffusion recommender model. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1370–1379. ACM, 2024. doi: 10.1145/3626772.3657825. URL https://arxiv.org/abs/2401.06982.

Bowen Zheng, Enze Liu, Zhongfu Chen, Zhongrui Ma, Yue Wang, Wayne Xin Zhao, and Ji-Rong Wen. Pre-training generative recommender with multi-identifier item tokenization. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2025. URL https://arxiv.org/abs/2504.04400.

Qiyong Zhong, Jiajie Su, Yunshan Ma, Julian McAuley, and Yupeng Hou. Pctx: Tokenizing personalized context for generative recommendation. *CoRR*, abs/2510.21276, 2025. URL https://arxiv.org/abs/2510.21276.

Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. S3-Rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*, pp. 1893–1902. ACM, 2020. doi: 10.1145/3340531.3411954.

Kun Zhou, Hui Yu, Wayne Xin Zhao, and Ji-Rong Wen. Filter-enhanced MLP is all you need for sequential recommendation. In *Proceedings of the ACM Web Conference 2022*, pp. 2388–2399. ACM, 2022. doi: 10.1145/3485447.3512111.

Sharare Zolghadr, Ole Winther, and Paul Jeha. Generative diffusion models for sequential recommendations. In *ROEGEN@RecSys'24: 1st Workshop on Risks, Opportunities, and Evaluation of Generative Models in Recommender Systems*, 2024. URL https://arxiv.org/abs/2410.19429.

A Model and Implementation Details

A.1 Dataset statistics

Table 3 reports the statistics of the preprocessed datasets described in Section 4.1.

Table 3: Preprocessed dataset statistics; Avg. len is the mean sequence length.

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

A.2 Compared models

Sequential recommenders. **GRU4Rec** (Hidasi et al., 2016) models interaction sequences with gated recurrent units. **BERT4Rec** (Sun et al., 2019) uses a bidirectional Transformer trained by masked-item prediction. **SASRec** (Kang & McAuley, 2018) instead uses causal self-attention for next-item prediction. **FMLP-Rec** (Zhou et al., 2022) replaces attention with MLP blocks and learnable filters that suppress noise in the frequency domain. **HSTU** (Zhai et al., 2024) introduces hierarchical sequential transduction units designed to scale to large recommendation systems. **FDSA** (Zhang et al., 2019) models feature and item-level transitions with separate self-attention streams. **S^3-Rec** (Zhou et al., 2020) pretrains a sequential encoder with self-supervised objectives over items, attributes, and subsequences.

Generative retrieval models. **TIGER** (Rajput et al., 2023) quantises item-text embeddings into semantic identifiers with an RQ-VAE and autoregressively generates the identifier of the next item. **LETTER** (Wang et al., 2024) augments this tokenisation process with collaborative signals and diversity regularisation. **ActionPiece** (Hou et al., 2025) learns context-dependent action tokens from co-occurrence patterns and uses permutation regularisation during generation.

Controlled baselines. **SASRec + T5** is our SASRec implementation with the same hybrid semantic–collaborative item representations as DARTS. **DreamRec** (Yang et al., 2023) generates a target-item representation by conditional diffusion and uses classifier-free guidance. **DiffuRec** (Li et al., 2024) corrupts the target-item embedding and trains a Transformer denoiser conditioned on the interaction history. **ADRec** (Chen et al., 2025) initialises diffusion from a pretrained collaborative item space and then optimises the sequential recommendation objective. For the three diffusion methods, the *+T5* variants replace their item tables with the same hybrid representations used by DARTS.

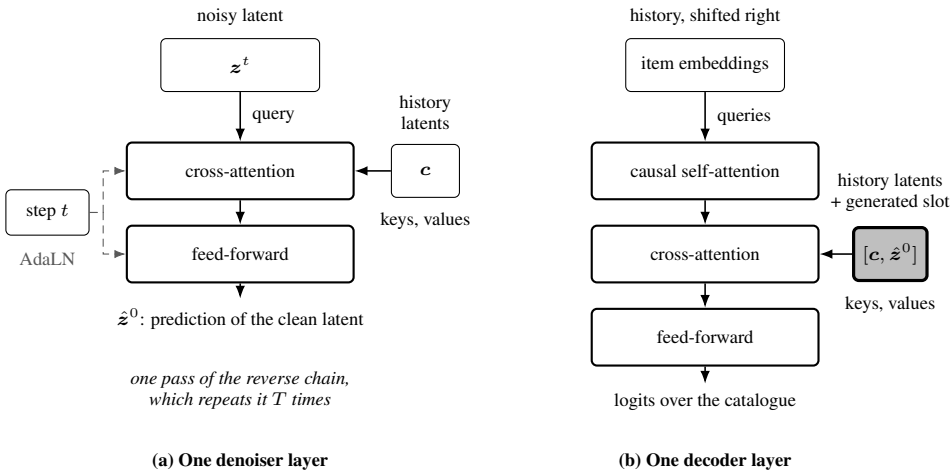
Result provenance. Results for GRU4Rec, BERT4Rec, SASRec, FMLP-Rec, HSTU, FDSA, S^3-Rec, TIGER, and LETTER are taken from Zheng et al. (2025). ActionPiece results are taken from Zhong et al. (2025). Both sources use the dataset preprocessing and evaluation protocol described in Section 4.1. We train SASRec + T5, DreamRec + T5, DiffuRec + T5, and ADRec + T5 ourselves in a shared codebase.

A.3 Module block diagrams

Figure 3 details one layer of the denoiser g_θ and of the decoder D_ψ introduced in Section 3.3. In both modules, the conditioning latents enter only as the keys and values of a cross-attention and are never added to the residual stream. The encoder E_ϕ is a stack of pre-norm bidirectional self-attention blocks with rotary position embeddings (Su et al., 2024).

Denoiser. Each block of g_θ is an adaLN-Zero block (Peebles & Xie, 2023) in which $\boldsymbol{z}^t$ cross-attends prefix states $\boldsymbol{c}$ (Equation 5).

Decoder. Each block of D_ψ applies causal self-attention with rotary position embeddings to the right-shifted history item representations $\boldsymbol{e}_i$, then cross-attends to a latent sequence of length L. This sequence is $E_\phi(\tilde{s})$ in Stage 1, and $[\boldsymbol{c}, \hat{\boldsymbol{z}}^0]$ in Stage 3 and at inference. The output at slot L is scored against the tied item matrix, which gives the catalogue logits $\boldsymbol{l}$.



(a) One denoiser layer **(b) One decoder layer**

Figure 3: One layer of each module. In both, conditioning arrives from the right as the *keys and values* of a cross-attention and is never added to the stream being transformed.

A.4 TRAINING STAGES

Table 4 summarises the parameters trained and frozen at each of the three stages of Section 3, together with their objectives.

Table 4: The three training stages. Each updates a disjoint parameter set while the others remain frozen. Only Stage 3 uses an item-level objective.

Stage	Trained	Frozen	Objective
1. Autoencoder	E_ϕ, D_ψ, $\boldsymbol{e}^{\text{id}}$	$\boldsymbol{e}^{\text{sem}}$	$\mathcal{L}_{\text{MLM}}$: catalogue CE on masked slots (Eq. 4)
2. Diffusion	g_θ	E_ϕ, D_ψ, $\boldsymbol{e}$	$\mathcal{L}_{\text{diff}}$: latent RMSE, $\boldsymbol{z}^0$-prediction (Eq. 7)
3. Decoder	D_ψ	E_ϕ, g_θ, $\boldsymbol{e}$	$\mathcal{L}_{\text{dec}}$: BPR on self-mined hard negatives (Eq. 8)

A.5 INFERENCE

At inference, we encode s^{mask} once to obtain $\boldsymbol{c}$. We sample a final latent by running the T reverse steps from Gaussian noise, then pass $[\boldsymbol{c}, \hat{\boldsymbol{z}}^0]$ to the decoder. The final logits rank the full catalogue. Our default evaluation uses one sample. Section 4.3 studies repeated samples that share the same conditioning prefix and differ only in their diffusion noise.

A.6 HYPERPARAMETERS

Table 5 lists the hyperparameters of the three training stages of Section 3. Histories are truncated to the 20 most recent interactions and left-padded to that length; the catalogue is extended with a padding symbol and a `[MASK]` symbol, whose logits are set to $-\infty$ before ranking. The frozen semantic table $\boldsymbol{e}^{\text{sem}}$ is obtained by encoding each item's title and description with the `sentence-t5-base` checkpoint (Ni et al., 2022) and projecting the result to d_v dimensions. All stages use a linear learning-rate schedule with warmup and gradient clipping.

Because each stage reloads the frozen components of its predecessor, the architectural settings in the upper block of Table 5 are necessarily shared across stages. Stage 1 selects checkpoints on masked-item accuracy, Stage 2 on validation latent RMSE, and Stage 3 on NDCG@10. Stage 2 is the longest stage in wall-clock terms despite its simple objective, because every validation pass runs the full T-step reverse chain; Stage 3 runs that chain on every training batch as well, which is why it uses far fewer epochs.

B EFFECT OF ITEM REPRESENTATIONS

Every model in Section 4.2 that we train ourselves uses hybrid semantic–collaborative item representations, in which a frozen T5 encoding of the item text is mixed with a learned identifier embedding. Because this choice is shared across DARTS and the diffusion baselines, it cannot explain the differences between them. It could, however, inflate the comparison against published numbers that use identifier embeddings only. Table 6 therefore reports DARTS and each baseline on SCIENTIFIC in both configurations. Each pair differs by a single configuration flag, with architecture, data, optimiser, and schedule held fixed. Our identifier-only SASRec reproduces the figure reported by Zheng et al. (2025), which both validates our implementation and confirms that our preprocessing and evaluation protocol match the source of the quoted baselines in Table 1.

Semantic features help every model, but the size of the effect varies by more than an order of magnitude and does not track the strength of the underlying model. DiffuRec, the strongest diffusion baseline, gains 23.2% NDCG@10, while DreamRec—the weakest—gains only 10.7% and remains far behind regardless of representation. ADRec is the extreme case, improving by 190.7%: with identifier embeddings alone it is barely competitive (NDCG@10 0.0067), and semantic features recover most of the gap to the other baselines. This is consistent with its design, which relies on a pretrained item table to warm-start the denoiser and is therefore unusually sensitive to how items are represented.

DARTS is the least sensitive of the four to this choice, gaining 14.4% NDCG@10, and it is the only one whose identifier-only configuration remains competitive with the baselines in their stronger set-

Table 5: Hyperparameters for the three training stages. Architectural settings are shared across stages because later stages reload the frozen components of earlier ones. Dashes indicate settings that do not apply to a stage.

		Stage 1 Autoencoder	Stage 2 Diffusion	Stage 3 Decoder
Architecture	Item embedding dimension d_v		64	
	Latent dimension d_z		64	
	Encoder layers / heads		2 / 2	
	Decoder layers / heads		2 / 2	
	Denoiser layers / heads	—	2 / 2	
	Dropout		0.1	
	Item mixing weight ω		0.5	
Diffusion	Timesteps T	—	200	
	Schedule	—	linear	
	$\beta_1 \rightarrow \beta_T$	—	$10^{-4} \rightarrow 2 \times 10^{-2}$	
	Sampler	—	ancestral, T steps	
Objective	Masking rate p	0.4	—	—
	Hard negatives n	—	—	300
	Temperature τ	—	—	0.07
Optimisation	Optimiser	AdamW	Adam	AdamW
	Learning rate	10^{-3}	10^{-4}	10^{-5}
	Weight decay	10^{-3}	0	0
	Batch size	1024	1024	1024
	Epochs	600	100	50
	Warmup ratio / grad. clip		0.05 / 1.0	

Table 6: Effect of hybrid semantic–collaborative item representations on SCIENTIFIC. Each model is evaluated with learned identifier embeddings only and with the hybrid T5–collaborative representations used throughout Section 4. The last column reports the relative change in NDCG@10.

	Identifier embeddings				Hybrid (+T5)				
Model	R@5	R@10	N@5	N@10	R@5	R@10	N@5	N@10	ΔN@10
SASRec	0.0259	0.0412	0.0150	0.0199	0.0277	0.0452	0.0177	0.0232	+16.6%
DreamRec	0.0104	0.0176	0.0067	0.0089	0.0112	0.0196	0.0071	0.0098	+10.7%
DiffuRec	0.0251	0.0393	0.0152	0.0197	0.0302	0.0468	0.0190	0.0243	+23.2%
ADRec	0.0077	0.0137	0.0048	0.0067	0.0236	0.0374	0.0150	0.0195	+190.7%
DARTS	0.0285	0.0446	0.0185	0.0236	**0.0329**	**0.0524**	**0.0207**	**0.0270**	+14.4%

ting: at NDCG@10 0.0236 it improves over the best identifier-only baseline by 18.6% and already matches SASRec + T5 (0.0232), falling only slightly short of DiffuRec + T5 (0.0243).

Two observations follow. First, semantic features are not a substitute for the generative formulation: enriching DreamRec's item representations does not make it competitive, the ordering of the diffusion baselines is unchanged by the switch, and DARTS retains most of its advantage without them. Second, the margins reported in Section 4.2 are measured against baselines in their stronger configuration, so they are not an artefact of the item encoder.

C Discrete Supervision Inside the Diffusion Process

Section 1 attributes the collapse of existing diffusion recommenders partly to supervising the denoiser with a loss defined on decoded item logits, and Section 4.4 summarises the experiment that isolates this mechanism. This appendix gives the setup and the full sweep.

Setup. We take a single Stage-1 checkpoint and retrain Stage 2 with the mixed objective

$$\mathcal{L}_\alpha(\theta) \;=\; (1-\alpha)\,\mathcal{L}_{\text{diff}}(\theta) \;+\; \alpha\,\mathcal{L}_{\text{CE}}(\theta), \qquad \mathcal{L}_{\text{CE}}(\theta) \;=\; -\,\mathbb{E}_{s,\,t,\,\boldsymbol{\epsilon}}\left[\log p_\psi\big(i_L \,\big|\, D_\psi\big([\boldsymbol{c}, \hat{\boldsymbol{z}}^0]\big)_L\big)\right], \tag{11}$$

where $\hat{\boldsymbol{z}}^0 = g_\theta(\boldsymbol{z}^t, t, \boldsymbol{c})$ is the denoiser's current prediction, decoded by the frozen decoder at every training step. Setting $\alpha = 0$ recovers the Stage-2 objective of Section 3.5, while $\alpha = 1$ supervises the denoiser with cross-entropy alone. Nothing else changes: same initialisation, same architecture, same schedule. Because the encoder and decoder stay frozen throughout, α is the only route by which a discrete objective can reach the generative stage, which is what makes the comparison clean.

Table 7: Adding catalogue cross-entropy to the Stage-2 objective on Scientific. Runs differ only in the mixing weight α of Equation 11: $\alpha = 0$ is our latent-only objective, while $\alpha = 1$ uses cross-entropy alone. Diagnostics are those of Section 4.3.

α	J	$R\uparrow$	PR$\uparrow$	NDCG@5
0.0	0.30	**0.300**	**4.70**	**0.0207**
0.1	0.43	0.140	1.50	0.0151
0.3	0.46	0.040	1.45	0.0111
0.5	0.52	0.018	1.02	0.0102
0.7	0.50	0.010	0.85	0.0098
1.0	0.52	0.005	0.65	0.0089

Results. Table 7 shows that the damage starts immediately and then compounds. Adding a tenth of a cross-entropy term already more than halves R, from 0.30 to 0.14, and cuts PR from 4.7 to 1.5, leaving barely more than a single sampled direction. Both keep falling as α grows: at $\alpha = 1$ the variance ratio is 0.005, sixty times smaller than at $\alpha = 0$, and PR reaches 0.65. Equation 10 gives PR ≥ 1 for any nonzero covariance. In practice, we compute each context's ratio as $(\operatorname{tr}\boldsymbol{C_c})^2/(\operatorname{tr}(\boldsymbol{C_c^2}) + \epsilon)$ with $\epsilon = 10^{-8}$, set it to zero when $\operatorname{tr}\boldsymbol{C_c} \leq 10^{-8}$, and report the average over contexts. Values below one therefore come from contexts whose samples have collapsed numerically: they are either counted as zero or, when $\operatorname{tr}\boldsymbol{C_c}$ is of order 10^{-4} or smaller, shrunk by ϵ. An average below one thus signals widespread collapse, but its magnitude should not be read as an effective dimensionality.

Ranking, which the discrete term is intended to improve, never exceeds the $\alpha = 0$ result and degrades monotonically: NDCG@5 is 27% lower already at $\alpha = 0.1$ and 57% lower at $\alpha = 1$.

The decoded-set dissimilarity J moves the other way, rising from 0.30 to roughly 0.52. Taken alone this would read as more varied recommendations, but it coincides with collapsing latent variation and collapsing accuracy, so the varied slates are not drawn from a meaningfully wider distribution: once the denoiser is trained through decoded logits, the decoder turns near-identical latents into different low-quality lists. J therefore reflects the sensitivity of the decoder as much as the breadth of the sampling, which is why we read it alongside R, PR, and NDCG rather than on its own.

Cross-entropy inside the diffusion process thus costs generative behaviour and ranking accuracy at the same time, which is the recommendation analogue of the argument Hu et al. (2026) make for language.

D Effect of Updating Item Representations During Diffusion Training

Appendix C studies the Stage-2 objective while keeping the item representations fixed. We now examine the second design choice: whether these representations should remain fixed or be updated during diffusion training.

Setup. Starting from the same Stage-1 checkpoint, we retrain Stage 2 under four configurations. We either freeze the item representations, as in Section 3.5, or update them jointly with the denoiser. For each setting, we train with either latent regression or catalogue cross-entropy on decoded logits, corresponding to $\alpha = 0$ and $\alpha = 1$ in Equation 11. All other components and hyperparameters are unchanged. The two frozen configurations are also reported in Table 7.

Table 8: The two design choices, ablated jointly on SCIENTIFIC. *Item representations* are either frozen after Stage 1 or updated during Stage 2 alongside the denoiser; the Stage-2 objective is either latent regression alone or cross-entropy on decoded logits ($\alpha = 1$ in Equation 11). The first row is DARTS. Diagnostics are those of Section 4.3.

Item representations	Stage-2 objective	J	$R\uparrow$	PR $\uparrow$	NDCG@5
Frozen	**Latent regression**	0.30	**0.300**	**4.70**	**0.0207**
Frozen	Cross-entropy	0.52	0.005	0.65	0.0089
Trained	Latent regression	0.25	0.024	1.30	0.0121
Trained	Cross-entropy	0.43	0.050	2.40	0.0107

Results. Table 8 shows that keeping the representations fixed is important even when no cross-entropy is used. Under latent regression, updating the representations reduces R from 0.300 to 0.024, reduces PR from 4.70 to 1.30, and lowers NDCG@5 by 42%. Changing only the objective to cross-entropy causes an even larger reduction when the representations remain frozen. The combination used by DARTS, frozen representations with latent regression, achieves the highest R, PR, and NDCG@5.

The effects are not additive. When the representations are updated under cross-entropy, $R = 0.050$ and $\mathrm{PR} = 2.40$, both higher than in the frozen cross-entropy configuration, although NDCG@5 remains 48% below DARTS. The experiment therefore establishes the effect of each design choice separately, but does not support an ordering among the three alternative configurations.

As in Appendix C, J does not track recommendation quality. It is lowest for the updated-representation regression run (0.25), whose NDCG@5 is 42% below ours, and highest for the two cross-entropy runs. We therefore interpret J only together with the latent diagnostics and NDCG.

E Stochastic Discriminative Baseline

A discriminative recommender already defines a categorical next-item distribution, even though standard inference sorts it into a deterministic ranking. We test whether randomising this distribution provides the same benefit as sampling learned latent predictive states. For SASRec's catalogue distribution $p_{\mathrm{SAS}}(i \mid \boldsymbol{c})$, draw independent Gumbel variables $g_i^{(m)}$ and form

$$q^{(m)}(i \mid \boldsymbol{c}) = \mathrm{softmax}_i\Big(\log p_{\mathrm{SAS}}(i \mid \boldsymbol{c}) + g_i^{(m)}\Big).$$

Sorting each perturbed vector gives a Plackett–Luce ranking; in particular, its first item is an exact categorical draw from p_{SAS}. More generally, given M predictive distributions $q^{(m)}(i \mid \boldsymbol{c})$, we

aggregate them with the generalised power mean

$$\hat{q}_{M,p}(i \mid \boldsymbol{c}) = \left(\frac{1}{M} \sum_{m=1}^{M} q^{(m)}(i \mid \boldsymbol{c})^p \right)^{1/p}, \qquad p \neq 0. \tag{12}$$

The limits $p \to 0$ and $p \to \infty$ give the geometric mean and max pooling, respectively, while $p = 1$ gives the arithmetic mean used for the posterior-predictive experiment in Figure 6. For this baseline, we use the checkpoint's training temperature 0.07 and apply Equation 12 to the perturbed SASRec distributions. This is a deliberately simple stochastic discriminative baseline: its variation is injected independently at the catalogue scores rather than learned as a distribution over predictive states.

Table 9: Post-hoc stochastic inference for SASRec + T5 on SCIENTIFIC. Gumbel noise is applied independently to every catalogue log-probability at temperature 0.07. We aggregate M perturbed distributions with power p; all powers coincide at $M = 1$. Deterministic SASRec sorts the unperturbed distribution.

Inference	M	p	R@5	R@10	N@5	N@10
Deterministic	–	–	0.0277	0.0452	0.0177	0.0232
Gumbel	1	–	0.0058	0.0104	0.0035	0.0050
Gumbel	5	0	0.0203	0.0343	0.0128	0.0173
Gumbel	5	1	0.0063	0.0123	0.0039	0.0058
Gumbel	5	2	0.0056	0.0109	0.0035	0.0051
Gumbel	5	∞	0.0055	0.0106	0.0034	0.0050

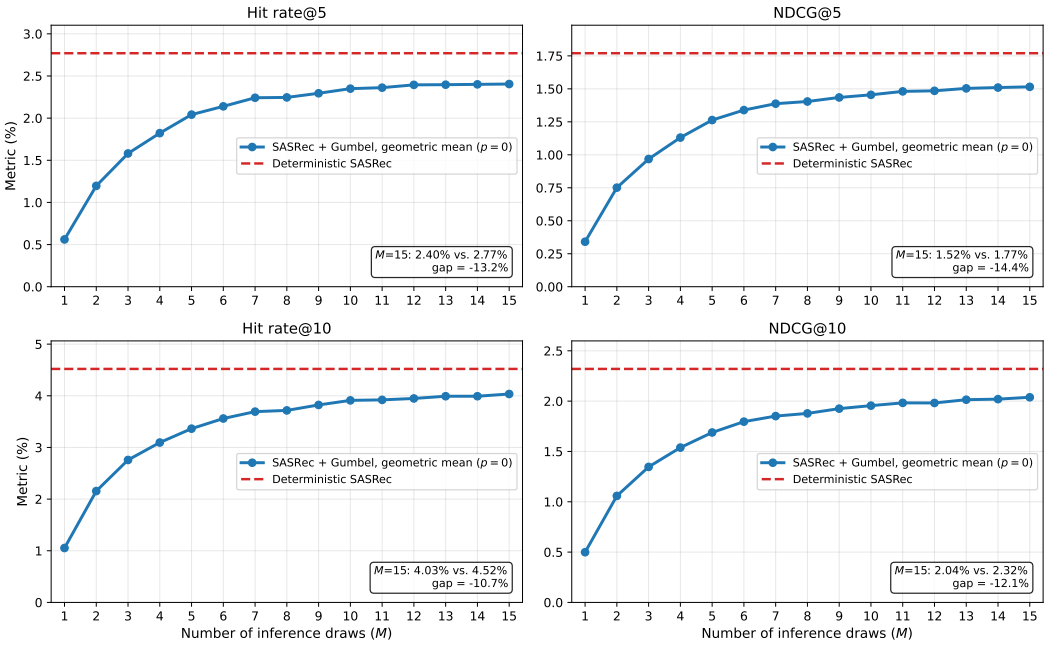


Figure 4: Stochastic SASRec on SCIENTIFIC: geometric-mean aggregation ($p = 0$) of M Gumbel-perturbed catalogue distributions at a fixed top-k cutoff. The dashed line is deterministic SASRec, which sorts the unperturbed distribution. Each panel is one metric; the inset boxes report the value at $M = 15$ and its gap to the deterministic ranking. Additional draws only recover part of the accuracy lost at $M = 1$ and saturate strictly below the deterministic ranking.

Table 9 shows that, for SASRec, output perturbation is not sufficient for useful stochastic recommendation. A single Gumbel ranking loses about 80% NDCG@5 and 79% NDCG@10 relative to sorting SASRec's distribution. Arithmetic aggregation ($p = 1$), which matches the posterior-predictive experiment in Figure 6, recovers little by $M = 5$ and remains about 78% and 75% below deterministic SASRec on NDCG@5 and NDCG@10. Larger powers perform similarly.

The geometric mean ($p = 0$) recovers much faster, but this does not indicate complementary predictions. Gumbel noise is additive in log-probability space, so geometric aggregation averages away the independent perturbations and converges back to the original SASRec ordering. At $M = 5$ it still remains about 28% below deterministic SASRec on NDCG@5 and 26% below it on NDCG@10. Figure 4 extends this sweep to $M = 15$: all four metrics saturate well before that point and stay 11–14% below the deterministic ranking, so further draws would not close the gap. Thus, the upward geometric-mean curve reflects removal of noise introduced at $M = 1$, whereas DARTS's latent samples improve beyond its own single-draw ranking. This negative control shows that stochasticity and aggregation alone do not explain our gains; they require draws with complementary, context-dependent predictive signal.

F Pooling Draws: UnionRecall

Section 4.4 spends M inference draws by averaging their item distributions and ranking once on the aggregate. The more obvious alternative is to spend them on the slate itself: run the model M times, take the top-n of each run, and show the user the union. This appendix reports that variant, both because it is the natural way to read a diverse sampler and because it does not work, which is what makes the aggregation result of Section 4.4 non-trivial.

Definition. For a context $\boldsymbol{c}$ with relevant set $G_{\boldsymbol{c}}$, let $L_{\boldsymbol{c}}^{(1)}, \ldots, L_{\boldsymbol{c}}^{(M)}$ be M independently sampled ranked lists and let

$$U_{\boldsymbol{c}} \;=\; \bigcup_{m=1}^{M} \text{top-}n\big(L_{\boldsymbol{c}}^{(m)}\big), \qquad \text{UnionRecall@}n(M) \;=\; \mathbb{E}_{\boldsymbol{c}}\left[\frac{\big|U_{\boldsymbol{c}} \cap G_{\boldsymbol{c}}\big|}{\big|G_{\boldsymbol{c}}\big|}\right]. \tag{13}$$

Under the leave-one-out protocol of Section 4.1 we have $|G_{\boldsymbol{c}}| = 1$, so the metric is the fraction of contexts for which *at least one* of the M draws places the held-out item in its top-n. At $M = 1$ it is the ordinary hit rate@n, and it is non-decreasing in M by construction. We report the mean pooled slate size $|U_{\boldsymbol{c}}|$ alongside it, which is the price paid: a deterministic model keeps $|U_{\boldsymbol{c}}| = n$ for every M, so any growth is a direct readout of sampling diversity.

Results. Table 10 gives both quantities on Scientific for DARTS. UnionRecall@5 rises from 0.0344 at $M = 1$ to 0.0547 at $M = 30$, a 59% relative increase, and the slate grows from 5 to 11.5 items. The union saturates well below $M \cdot n$: thirty lists of five items contain 11.5 distinct items between them rather than 150, so the draws are diverse but concentrated on a stable neighbourhood of the catalogue rather than wandering. The same shape holds at $n = 10$. Read as a diversity diagnostic this is informative, and it is the item-space counterpart of the latent measurements in Section 4.3: only a model that genuinely samples can grow $|U_{\boldsymbol{c}}|$ at all.

Table 10: UnionRecall@$n(M)$ and mean pooled slate size $|U_c|$ on Scientific, as a function of the number of independent inference draws M. $M = 1$ recovers the ordinary hit rate@n. A model whose draws were identical would keep $|U_c| = n$ and a flat recall for every M.

M	1	2	3	4	5	10	15	20	25	30
UnionRecall@5	0.0344	0.0383	0.0411	0.0434	0.0447	0.0490	0.0514	0.0530	0.0539	0.0547
mean $\lvert U_c\rvert$	5.00	6.29	7.06	7.62	8.04	9.38	10.16	10.72	11.15	11.49
UnionRecall@10	0.0545	0.0596	0.0636	0.0660	0.0681	0.0731	0.0759	0.0777	0.0791	0.0803
mean $\lvert U_c\rvert$	10.00	12.31	13.65	14.61	15.36	17.64	18.95	19.87	20.59	21.17

Why this is not an accuracy gain. It is tempting to compare UnionRecall@$5(M)$ against hit rate@5 and report the difference as an improvement, but the two are not measured at the same budget: pooling M draws shows the user $|U_{\boldsymbol{c}}| \geq 5$ items. The fair comparison holds the number of items shown fixed, which means comparing UnionRecall@$n(M)$ at mean slate size $|U_{\boldsymbol{c}}|$ against the single-draw hit rate@k at $k = |U_{\boldsymbol{c}}|$. Figure 5 plots both curves on that common axis.

The single-draw curve dominates throughout. At $M = 5$, pooling the top-5 of five draws shows 8.0 items on average and recovers the target for 4.47% of users, while simply extending one draw to its

top-8 shows the same number of items and recovers 4.62%. The gap is small but consistent, and it has the same sign at $n = 10$. Diversity, in other words, is not by itself a source of accuracy: a longer slate from one draw uses the budget slightly better than a pooled slate from several.

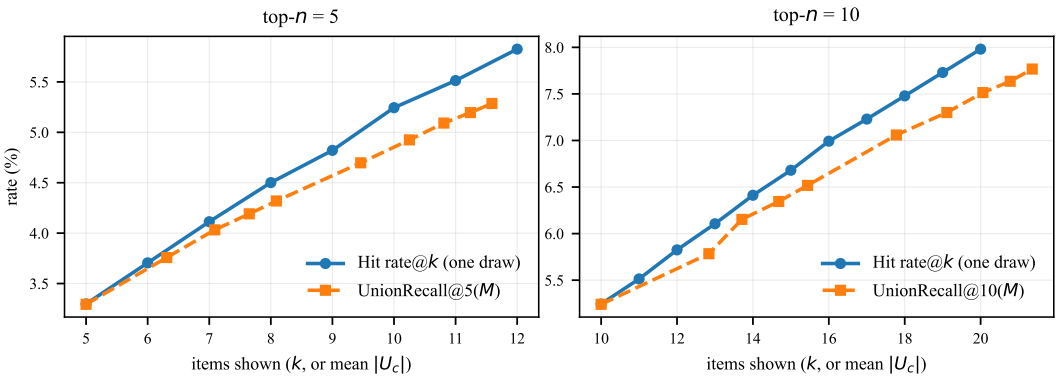


Figure 5: UnionRecall@$n(M)$ vs single-draw hit rate@k on SCIENTIFIC, both against the number of items shown (the UnionRecall curve sweeps M from 1 to 30, each point at its mean slate size $|U_c|$). At matched slate size a single longer list matches or beats pooling several draws: sampler diversity is not by itself a source of accuracy.

This is why Section 4.4 aggregates in probability space rather than in item space. Averaging the per-draw distributions and then truncating uses the M draws to sharpen the estimate of $p(i \mid \boldsymbol{c})$ while leaving the slate at k items. The comparison against the $M = 1$ model is therefore budget-neutral, and the extra draws improve performance by roughly 3% (Figure 6). The distinction matters for how the claim should be read: the value of a genuinely generative recommender here is a better posterior estimate at fixed slate size, not a longer list.

G INFERENCE COST

Section 4.4 shows that aggregating multiple independently sampled predictions can further improve ranking performance. This comes at an additional inference cost, since each draw requires a complete reverse diffusion process. We therefore measure the serving cost of DARTS and the two baselines that use the same item representations.

Protocol. We report throughput as the number of complete predictions processed per second. A prediction includes history encoding, reverse diffusion, decoding of the final position, and catalogue scoring. All models use a latent width of 64, a catalogue of 25,848 items, and left-padded histories of length 50. Measurements are performed at batch size 1024 on a single NVIDIA V100 (32 GB).

DARTS uses 200 reverse diffusion steps and DiffuRec uses 32 steps (as stated in the paper), matching the configurations used for the results in Table 1. For $M > 1$, one complete prediction comprises M independently sampled reverse chains and their aggregation using Equation 12 with $p = 1$.

Comparison with DiffuRec. At a single draw ($M = 1$), DARTS achieves a throughput of 2,102 predictions per second, compared with 590 for DiffuRec+T5, while achieving an 11.1% higher NDCG@10 as reported in Table 1. Thus, relative to DiffuRec+T5, DARTS is both faster and more accurate at the single-sample operating point.

This result is due to the substantially lower cost of each reverse step in our denoiser. DARTS cross-attends from a single latent query token to the encoded history, resulting in a per-step latency of 2.4 ms at this batch size. In contrast, DiffuRec re-encodes the complete item sequence with a four-block transformer at every diffusion step, requiring 54 ms per step. The lower per-step cost therefore more than compensates for the 6.25× larger number of reverse steps in DARTS.

Cost of additional samples. Increasing M requires repeating the complete reverse diffusion chain and therefore reduces throughput approximately proportionally to the number of sampled trajecto-

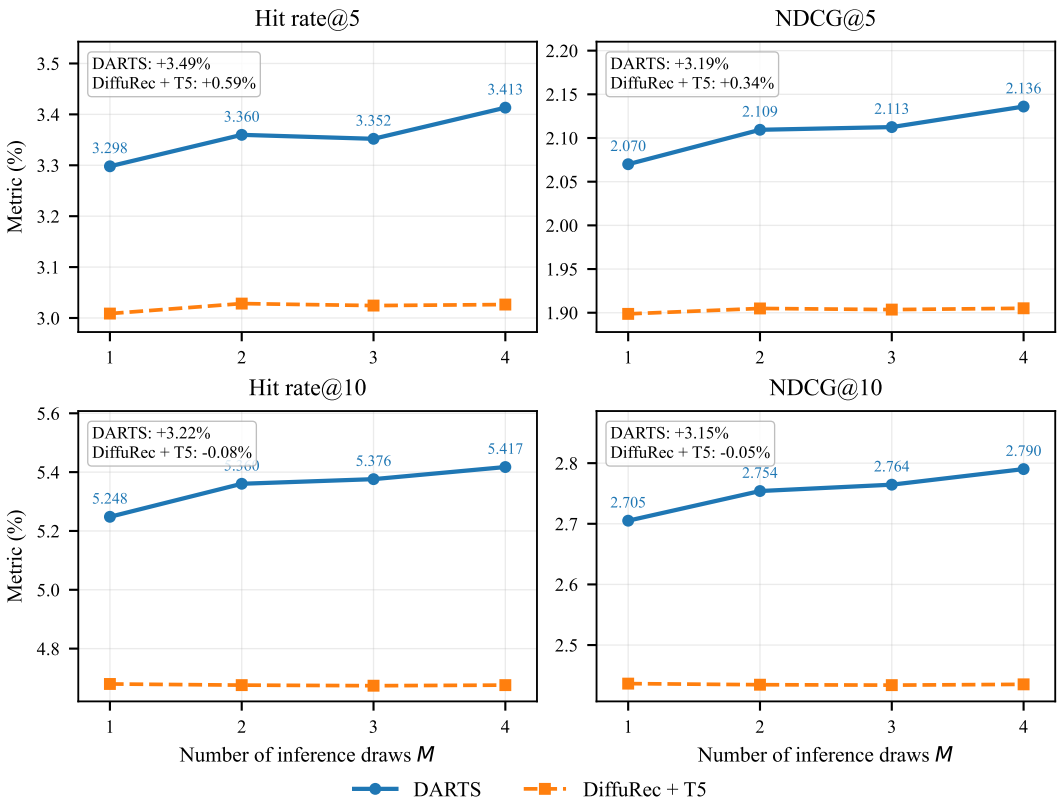


Figure 6: Posterior-predictive aggregation on SCIENTIFIC: ranking by the arithmetic mean ($p = 1$) of M independent draws at a fixed top-k cutoff. Each panel is one metric; the inset boxes report the relative change from $M = 1$ to $M = 4$ and the point labels give DARTS values in percent. DARTS turns extra draws into accuracy on every metric; DiffuRec, whose draws nearly coincide, does not.

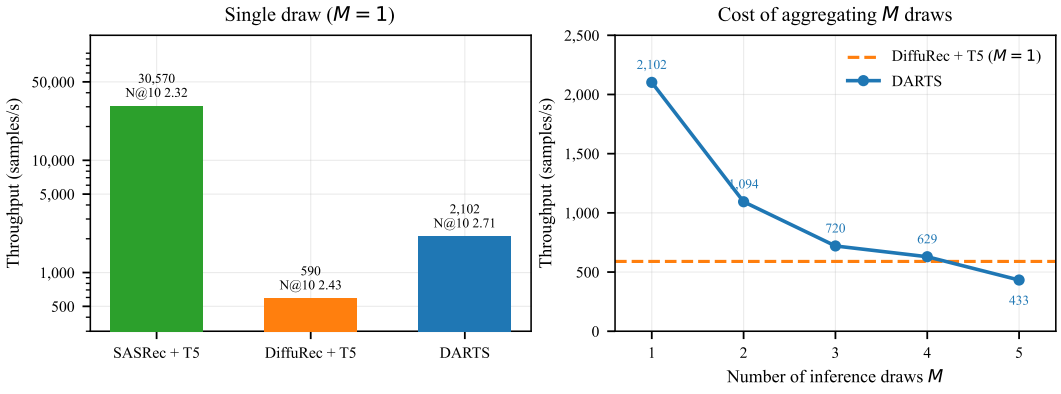


Figure 7: End-to-end inference throughput on SCIENTIFIC at batch size 1024 on a single NVIDIA V100 (32,GB). *Left:* single-draw ($M = 1$) throughput on a log scale, with NDCG@10 annotated. *Right:* throughput as the number of aggregated draws M increases. DARTS is faster than DiffuRec at $M = 1$ despite using more reverse diffusion steps, while additional draws incur an approximately linear serving cost.

ries. The right panel of Figure 7 compares the cost of aggregation with the $M = 1$ DiffuRec+T5 baseline. We use this reference because the samples produced by DiffuRec show substantially less variation and aggregation provides little additional ranking benefit (Figure 6).

For DARTS, increasing M from 1 to 4 improves NDCG@10 by 3.2% but reduces throughput from 2,102 to 629 samples per second, corresponding to a $3.3\times$ reduction. At $M = 4$, its throughput remains slightly above the 590 samples per second of DiffuRec+T5; at $M = 5$, it falls below this value.

These results characterize M as an explicit quality–compute trade-off rather than a requirement of the proposed approach. In particular, the diversity reported in Section 4.3 is obtained from a single sample ($M = 1$); additional draws can be used when the incremental ranking improvement justifies the additional serving cost.

Comparison with a non-generative recommender. SASRec+T5 requires only a single forward pass and reaches 30,570 samples per second, which is $14.5\times$ the throughput of DARTS at $M = 1$. This higher throughput comes with a 14.2% lower NDCG@10 in the reported comparison.

Thus, iterative generation introduces a substantial serving-cost premium relative to a single-pass sequential recommender. This comparison provides the computational context for the proposed generative approach: the additional inference cost enables stochastic sampling and aggregation, while the comparison with DiffuRec shows that the proposed latent denoiser can substantially reduce the per-step cost of diffusion-based recommendation.