DIFFUSION GENERATIVE RETRIEVAL OF ORDERED SLATES

Anonymous authors
Paper under double-blind review

ABSTRACT

In generative retrieval, the model memorizes the index in its own parameters. In this setting, the recommender stores the catalog in its weights and emits each item directly as a tuple of semantic ID tokens, without an inference time candidate pool. Diffusion is a natural decoder when the entries of a list are chosen jointly. Earlier diffusion approaches operate along the identifier levels of a single item and assemble the slate from separate decodes. **Diff**usion Generative **R**etrieval of **O**rdered **S**lates (DiffROS) transposes this axis. It denoises a canvas that holds the full slate, with a slot for every position. At each identifier level, DiffROS predicts every slot in the context of all others. The standard offline objectives, however, are modular set functions. They score each item separately and add those scores across the slate. Ranking items by their marginals, or separate contributions, therefore already maximizes these objectives. Conditioned on retrieval, DiffROS orders the recovered items several times more sharply than beam search and the autoregressive slate decoder. A next item decoder read out by beam search recovers more of the target items, while DiffROS better orders the ones it recovers. At matched capacity and with a slate of one hundred, we found that DiffROS recovers close to three times as many targets as an autoregressive slate decoder while its decoding depth remains constant in slate size, with three forward passes rather than three hundred.

1 INTRODUCTION

Recommenders increasingly return a slate of m items, most often through generative retrieval (Rajput et al., 2023; Deng et al., 2025; Ju et al., 2025; Hou et al., 2026). Because the slots in a slate interact, there is a case for decoding the whole slate as a single object. Existing work develops this idea through autoregression across slots, which decodes one slot after another. OneRec flattens the slate and decodes it item after item (Deng et al., 2025; Zhou et al., 2025). OGR decodes identifiers position by position under exposure order supervision (Hu et al., 2026). A family of generative rerankers applies list level objectives to related decoders that have sequential compute cost which grows with slate length (Lin et al., 2025b; Bello et al., 2018; Feng et al., 2021; Ren et al., 2024; Deffayet et al., 2023; Liu et al., 2021).

Diffusion is a natural decoder for an object whose parts are selected jointly. Within generative retrieval, however, prior work has primarily developed diffusion along another axis. DiffGRM, LLaDA-Rec, and MDGR denoise the identifier tokens of a *single* next item and then assemble a list from beams (Liu et al., 2026; Shi et al., 2025; Mu et al., 2026). This is a sound starting point because it keeps the decoder small and inherits the next item training signal already established in the field. It also leaves the slate axis open.

We work with the full canvas of slots, take its transpose, and diffuse it by levels. One prior system already reaches this axis and deserves emphasis. MaskGR uses masked diffusion over semantic IDs to decode two items in parallel, and evaluates it by averaging recalls computed for each item (Shah et al., 2025). This statistic gives (a, b) and (b, a) the same value, so the evaluation does not expose the effect of decoding the items jointly.

Two closely related systems study a different question. UniRank (Jia et al., 2026) applies masked diffusion across slate positions, uses full bidirectional attention, and presents a taxonomy of decoding schemes that subsumes ours. It scores a supplied candidate pool by dot product instead of

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.

decoding the vocabulary, so every item it emits belongs to that pool by construction. Prompt-to-Slate instead diffuses continuous embeddings and determines identity with nearest neighbor lookup (Tomasi et al., 2024); its model therefore never emits an identifier. Our contribution is the open catalog setting, where validity, duplication, and order become measurable, together with a decoder that operates in this setting. To our knowledge, it is the first diffusion based decoder to perform generative retrieval of an ordered slate of semantic IDs over an open catalog.

Evaluation is the harder half of the problem. Every system named above is evaluated with objectives that sum terms computed for each item, and such objectives are maximized by selecting items from their marginals alone. This is the probability ranking principle (Robertson, 1977; Gordon & Lenk, 1992; Chen & Karger, 2006). Section 3 restates its finite cutoff form because it predicts the measurements that follow.

Two evaluation findings accompany this result. First, the usual order aware metric, as commonly implemented, gives credit to a repeated correct prediction at every rank where it appears. When we instead use a variant that credits each target once, the ranking reverses and the other decoder appears to win. This result belongs to a broader line of reality checks on offline evaluation (Krichene & Rendle, 2020; Ferrari Dacrema et al., 2019; Li et al., 2021; Zhang et al., 2026a; Benigni et al., 2026; Volodkevich et al., 2024). Second, the decoders differ in how they arrange items rather than which items they select. Conditioned on retrieval, we found that DiffROS orders recovered items several times more sharply than beam search, relative to a measured shuffle null (Section 4.4).

2 DiffROS: Masked Diffusion over a Slate

DiffROS operates on a canvas as a masked discrete diffusion decoder, and it uses full bidirectional attention (Figure 1). Every position begins masked. On each forward pass, the model predicts all positions simultaneously. It commits a subset and returns the remaining positions to the mask. The commit scope defines this subset. We write T for the number of denoising steps and use $T \geq 1$ with level scope. Thus, one pass or more commits a complete identifier level across all slots. Confidence determines the commitment order only when $T > 1$ (Kim et al., 2026; Bansal et al., 2026). An optional remask repair procedure then runs for r rounds. It reopens any slot that contains an already emitted identifier and predicts the slot again while excluding that identifier (Ren et al., 2026). Throughout decoding, the model cross attends to a frozen request encoding. The model itself serves as the index, and no downstream model component resolves its output to an item (Tomasi et al., 2024). Consequently, an emitted identifier may name no item, while a duplicate consumes a slot without adding an item. We measure and report them in Section 4.

We represent a slate of m items as a canvas. Each item has a residual quantized semantic identifier with C levels, and each level has a codebook containing a few thousand entries. The resulting canvas is an $m \times C$ grid, and we flatten it in row major order to obtain mC positions. As an example, herein, on the internal corpus, $C = 3$; on the Amazon categories, $C = 4$, using an RQ-VAE tokenizer following Rajput et al. (2023), with three residual levels and a fourth token that disambiguates collisions, trained with AdamW at a learning rate of 10^{-3}.

The canvas presents two distinct axes: levels and slots. The chosen tokenizer in this study makes levels residual by construction, and parallel prediction therefore conflicts with it. This produces a factorization error that is structural, not a scheduling artifact (Yoo et al., 2026; Yao, 2026). Slots present a different case. They are exchangeable except for the order imposed by the target. Diffusion decoders for generative retrieval have so far parallelized the level axis (Liu et al., 2026; Shi et al., 2025; Mu et al., 2026). Related work accommodates this choice with non residual parallel codes (Hou et al., 2025; Lin et al., 2025a) or with decoding schedules that allocate steps unevenly across depth (Borsos et al., 2023; Kumbong et al., 2025). In DiffROS, we instead preserve the level axis and parallelize the slot axis.

DiffROS specifies a distribution over slates through a forward process that masks the canvas and a reverse process that reconstructs it (Austin et al., 2021; Sahoo et al., 2024; Ou et al., 2024; Nie et al., 2025). The canvas has two axes, so the forward process assigns a distinct rate to each identifier level. We have x_0 for the clean canvas and x_t for its partially masked version at noise level $t \in [0, 1]$, and $x^{j\ell}$ for the entry at slot $j \in \{1, \ldots, m\}$ and level $\ell \in \{1, \ldots, C\}$. Let u denote the encoded request and θ the decoder parameters and $\pi_\ell : [0, 1] \to [0, 1]$ denote this rate, nondecreasing with $\pi_\ell(0) = 0$

and $\pi_\ell(1) = 1$:

$$q_{t|0}(x_t \mid x_0) \;=\; \prod_{j=1}^{m}\prod_{\ell=1}^{C} q_{t|0}\Big(x_t^{j\ell} \mid x_0^{j\ell}\Big), \qquad q_{t|0}\Big(x_t^{j\ell} = \mathrm{M} \mid x_0^{j\ell}\Big) \;=\; \pi_\ell(t), \tag{1}$$

Thus, each position is independently preserved or replaced by the mask token M. Setting $\pi_\ell(t) = t$ for every level recovers the isotropic process used for language (Nie et al., 2025). By contrast, setting $\pi_\ell(t) = \mathrm{clip}\,(Ct - (C - \ell),\, 0,\, 1)$ masks levels sequentially. Its reverse is exactly the level scoped commit used during decoding, where level ℓ is resolved across all m slots before level $\ell + 1$ begins. Commit scope thus selects the forward process inverted by the sampler; it is not an unconstrained decoding heuristic. Our training masks are not sampled from either schedule above. Instead, they arise from on policy coherent noising, as described in Appendix C.

Two properties transfer from the isotropic case, and we use both. The conditional estimated by the network depends on x_t only through its unmasked entries, whose values equal those in x_0. It is therefore independent of t, and the mask predictor requires no time input (Ou et al., 2024). Moreover, with $\pi_\ell(t) = t$, the masked cross entropy is

$$\mathcal{L}(\theta) \;=\; -\,\mathbb{E}_{t,\,(u,x_0),\,x_t}\left[\frac{1}{t}\sum_{j=1}^{m}\sum_{\ell=1}^{C}\mathbf{1}\Big[x_t^{j\ell} = \mathrm{M}\Big]\log p_\theta\Big(x_0^{j\ell} \mid u,\, x_t\Big)\right] \tag{2}$$

This objective upper bounds $-\mathbb{E}\log p_\theta(x_0 \mid u)$, the negative log likelihood of the slate conditioned on the request (Sahoo et al., 2024; Ou et al., 2024; Nie et al., 2025). We do not sample training masks from Equation (1). Following Liu et al. (2026), we first run a single pass over the fully masked canvas x_1 to obtain a predictive distribution at each position. We define the difficulty of position (j, ℓ) as $\delta^{j\ell} = 1 - \max_v p_\theta(v \mid u, x_1)$, where v ranges over the codebook entries at that level. Let σ rank the mC positions from most to least difficult. View k then masks the ν_k most difficult positions,

$$\mathcal{M}^{(k)} \;=\; \{\sigma(1), \ldots, \sigma(\nu_k)\}, \qquad \nu_1 < \cdots < \nu_R, \tag{3}$$

which yields nested masked sets and progressively more visible context. The objective averages the masked cross entropy across views, $\mathcal{L} = \frac{1}{R}\sum_{k=1}^{R}\mathcal{L}^{(k)}$, where R is the number of views and we use $R = 4$. The scheme has two features that matter below. First, the model selects the masked set, so the mask distribution depends on θ. Second, the relevant coverage grows with the canvas. An mC position canvas contains $mC\,2^{mC-1}$ distinct target and context pairs, of which random masking reaches a vanishing fraction.

The decoders are trained against Equation (2), subject to two departures that affect its interpretation. Our masking draws a fixed count per view rather than an independent coin per position. In addition, our default loss reduction weights positions equally rather than weighting views by $1/t$. This choice trades the bound for a redistribution of gradient across mask ratios. Appendix C presents the reverse process, the bound and both departures; Appendix D gives the quantity estimated by each of the three factorizations.

The same distinction determines the scope of each commit, or how broadly the commit acts. At matched compute, committing one identifier level at a time across all slots dominates globally committing arbitrary canvas positions. We think this behavior is also dependent on the residual-based tokenizer and might be different if other method would be used. On the internal corpus, we measured a loss of about 15% of recall for item scoped commits and about 6% for globally scoped commits. On Amazon at $m = 3$, the same substitution is also worse. This effect belongs to the diffusion schedule alone, because level order is inherent to a causal decoder.

The item isolated factorization is $I_m \otimes \mathrm{tril}(C)$, and its critical path has C function evaluations. The autoregressive factorization across the slate is $\mathrm{tril}(mC)$, and its path has mC evaluations. DiffROS has a path of C, or $4C$ under three rounds of remask repair.

The item isolated decoder contains no slot structure. A single beam search over one item identifier prefix constructs the slate from its m highest scorings, and beam width affects parallel width rather than critical path length. This baseline is strong by construction. We train each slot independently against its own slate target, so beam search over the resulting conditional approximates the marginal over slate membership. Section 3 shows that the standard metrics reward this marginal.

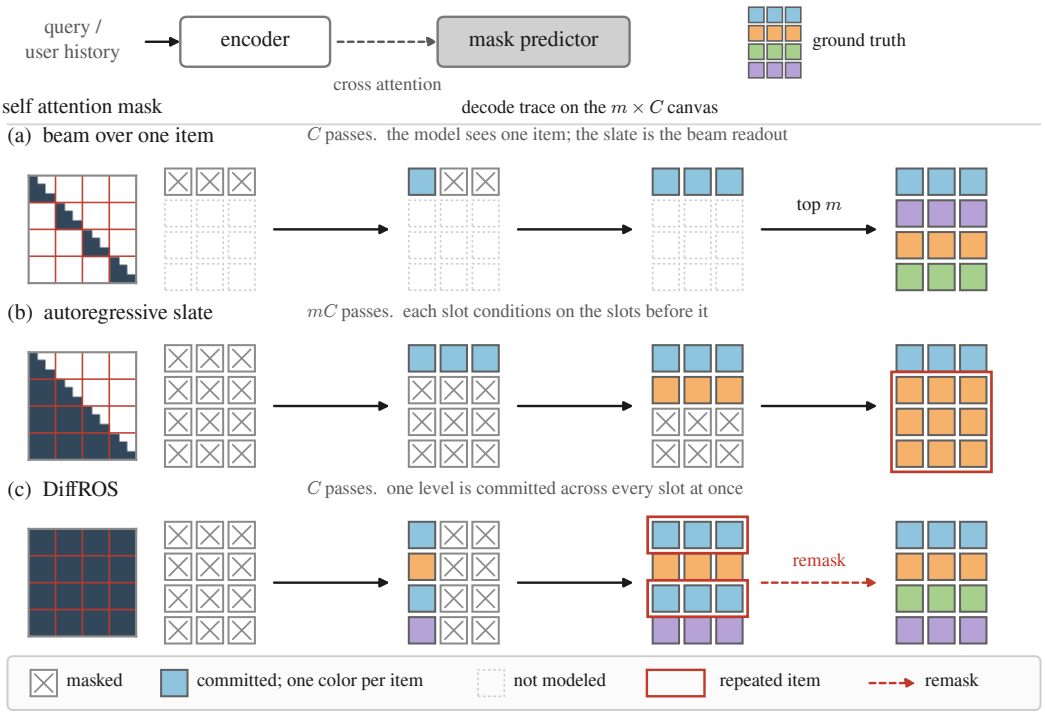

Figure 1: Three factorizations of one $m \times C$ canvas at $m = 4$ and $C = 3$. Rows are slots, and columns are identifier levels. Each color denotes one item, and the reference slate at the top right gives the target order.

At $T = 1$ with level scope and no repair, the sampler implements an exact reverse transition whose distribution factorizes across levels. Let $x_0^{:,<\ell}$ denote all positions at levels below ℓ, across every slot. Each pass produces one categorical distribution per position. We sample the m codes at level ℓ independently, without a uniqueness constraint or shared randomness. This independence is required for the product to define a valid distribution:

$$p_\theta(x_0 \mid u) \;=\; \prod_{\ell=1}^{C} \prod_{j=1}^{m} p_\theta\Big(x_0^{j\ell} \;\Big|\; u,\; x_0^{:,<\ell}\Big)\,. \tag{4}$$

Viewed level by level, this distribution forms a chain of C blocks. This is the stated depth property. Within each block, the slots are conditionally independent given the request and committed lower levels. This independence follows because the model defines a product over slots at that level and the decoder samples accordingly. Bidirectional attention thus provides conditioning on the lower levels of every slot, but no conditioning between slots at the same level. Repair breaks this factorization because the selector in Appendix D conditions on the realized sample and couples slots. At $T > 1$ the decoder commits a subset of a level at a time, so positions committed later condition on positions committed earlier at the same level, and the subset chosen is itself random. The intermediate canvases are then latent variables, and marginalizing them replaces the product by a sum of products, which no longer factorizes across slots.

3 Modularity of the Standard Objectives

Which of these factorizations wins is decided by the objective rather than by the decoder. The two objectives commonly reported in this field share a property that determines what they can measure. This property is *modularity*. For both recall@K and nDCG@K, the expected objective is a sum with one term for each selected item, and each term depends only on that item. The objective therefore cannot depend on which items appear together. Two consequences follow. The optimal set of size K contains the K items with the largest per item weights, and any dependence among

the targets is thus integrated into a univariate marginal before the objective observes it. The rank discount in $\mathrm{nDCG@}K$ merely reweights these per item terms, so it provides sensitivity to sorting but not to coupling. Appendix B gives more information.

This result is the probability ranking principle. Robertson (1977) derives it under exactly such a modularity assumption on utility. Chen & Karger (2006) show that it is optimal precisely for modular metrics and exhibit the nonmodular objectives under which it fails, while Gordon & Lenk (1992) establish the formal conditions. These known results concern ranking policies over scored candidates. We use them to predict and then measure the behavior of a decoder with no candidate set. Here, the ranking policy is instead a decoding procedure over an open catalog. Exact scoring of the item marginal for the top K items is Bayes optimal for both metrics. Beam search over identifier prefixes is a lossy realization of this scoring because prefix pruning can discard a positive item when its prefix probability falls below the beam (Yang et al., 2026). Evaluating a joint decoder by whether it beats beam on these metrics does not test the mechanism that the decoder was designed to express.

The same marginal that these objectives reward is what the next item baseline estimates. Trained with the item isolated mask and no slot embedding, every slot receives identical context, so that decoder reduces to maximum likelihood estimation of a single categorical over items, fitted with m samples per request, and a beam of width m returns its top m modes. Appendix D gives the derivation. The baseline is therefore aligned with the evaluation by construction, and it would beat joint slate factorization on both metrics.

4 Experiments

4.1 Corpora and Target Construction

We use Amazon Reviews 2023 (Hou et al., 2024) as the public benchmark, and include three of its categories: CDs and Vinyl, Video Games, and Industrial and Scientific. We follow the preprocessing of Shi et al. (2025). We key items on their parent identifier, keep only each user's earliest interaction with a repeated item, and apply iterative bilateral five core filtering to a fixed point. We use a leave N out split, where each user's last N interactions form the test slate. These slates are synthetic because the consecutive purchases were never displayed together, so we make no claim about structure across slots from this corpus. We evaluate users with at least $3N + 1$ interactions because training requires this. This choice restricts the evaluation to heavier users and limits the slate size. We include only $m = 1$ and $m = 3$ for this corpus.

We use a large scale search retrieval corpus of users and items as the second corpus. Its target consists of the top m candidates from a teacher ranker among the candidates that carry a semantic identifier. Because this is a distillation target, we make no claim about accuracy or preference from this corpus. It does support a controlled comparison. The comparison decoders share a tokenizer, encoder, training window, and parameter count, and their parameter counts differ by under a hundredth of a percent. Only the factorization varies. An emitted identifier becomes an item only through lookup in an inverted index. A distinct count therefore counts identifiers and provides an upper bound on item diversity.

We derived two target constructions for distinct experiments. The *natural* construction takes the corpus directly and uses the teacher's top m in the teacher's order. We use it for the scaling and compute cost measurements. The *bimodal* construction selects the best $m/2$ and worst $m/2$ candidates, and it places the best half before the worst half to create a deliberate discontinuity at the midpoint. This construction gives an order instrument that is sensitive to a detectable order signal. On a top m target, adjacent ranks induce nearly identical conditionals. Here, the first and last slots differ maximally, by roughly a factor of four in mean teacher relevance. The construction is deliberately favorable to a decoder with slot structure. It is not favorable under the standard metrics because both halves remain targets. An order free set metric scores the decoders identically regardless of emitted order, so the discriminator is order sensitive.

We key each construction in two ways: by the teacher's relevance score and by the placement that the production stack assigns to the candidate. They therefore cover a different population, and we interpret only comparisons within each table. Order statistics of this kind have precedent in set ordering, where the set is given and selection does not arise (Giuliari et al., 2023). The primary

Table 1: At $m = 30$ two bimodal constructions were used as targets keyed on relevance ($\mathcal{T}_{\text{rel}}$) and on placement ($\mathcal{T}_{\text{pos}}$). The last three columns report the decoder's own half split gap at the first two identifier levels and at an exact identifier match. (A larger gap means more ordering).

Target	Arm	Recall	dist	NFE	p_1 gap	p_2 gap	p_3 gap
$\mathcal{T}_{\text{rel}}$	Beam over one item	.085372	30.00	3	$+0.323$	$+1.049$	$+2.333$
	Autoregressive slate	.013840	2.42	90	$+0.061$	$+0.261$	$+0.432$
	DiffROS (r=0)	.017602	6.28	3	$+2.258$	$+3.568$	$+6.292$
	DiffROS (r=3)	.029734	20.97	12	$+2.301$	$+3.686$	$+8.645$
$\mathcal{T}_{\text{pos}}$	Beam over one item	.096243	30.00	3	$+0.258$	$+1.074$	$+1.825$
	Autoregressive slate	.012573	2.33	90	$+0.024$	$+0.210$	$+0.268$
	DiffROS (r=0)	.014409	7.76	3	$+2.945$	$+3.592$	$+5.421$
	DiffROS (r=3)	.026195	23.32	12	$+2.965$	$+3.574$	$+8.155$

Table 2: Amazon categories at $m = 3$, reporting recall at cutoff m. Last column as a soundness gate, DiffROS's recall as a multiple of the autoregressive decoder's recall at $m = 1$.

		$m = 3$		$m = 1$
Category	n_{test}	Autoregressive slate	DiffROS	$\times$ AR
CDs and Vinyl	42,768	.00241	.00253	1.01
Video Games	21,048	.00554	.00455	1.17
Industrial and Scientific	10,253	.00280	.00228	1.03

order instrument uses a half split. For each decoder, we partition its emitted slate at its own midpoint, average the target indices of recovered items on each side, and take their difference. A decoder that reproduces the target order reads low and then high. We score the instrument at the first identifier level and interpret it only as a within decoder contrast or a cross decoder ratio. Appendix A lists every instrument and its qualification, grouped by modularity in Table 5.

4.2 Results on Standard Metrics

We first compare all four decoders on standard ranking metrics across both corpora with same decoder depth and measure recall at thirty. The resulting ordering is unambiguous and disfavors the joint decoder, since beam over one item leads every slate factorization while also matching the lowest sequential depth of the four (Table 1).

For a single item, using a matched evaluator and no beam, DiffROS reaches parity with the autoregressive decoder and has zero duplicate rate (Table 2). We interpret this result as it supports that the tokenizer, encoder, and single item decoding are sound. At $m = 3$, the autoregressive decoder leads on two of the three categories and DiffROS leads on the third.

Next we test scalability in capacity and slate size. We first test whether the ordering survives additional decoder capacity. We compare the decoders across decoder depths of ten, twenty and forty layers while holding slate size fixed. As Figure 2 shows, the autoregressive decoder gains almost nothing beyond twenty layers, whereas both diffusion decoders continue to improve.

We next ask whether the slate axis becomes more valuable as the slate grows. Because the Amazon categories dataset cannot accommodate this question, we vary m over a tenfold range on the internal corpus (Table 3). The table supports two distinct comparisons. DiffROS versus the autoregressive decoder isolates factorization because neither decoder deduplicates, meaning that neither removes repeated identifiers. The remasked versus unremasked comparison instead isolates deduplication.

For per request hit rate, which is not constrained by the duplicate ceiling, the factorization margin increases monotonically across the sweep, while the deduplication margin remains flat. Both margins appear to increase for recall at m, but we interpret this divergence as evidence for the ceiling, not as a second deduplication result. In particular, the factorization margin grows against a decoder that no longer responds to slate size. At a hundred slots, the autoregressive decoder returns between

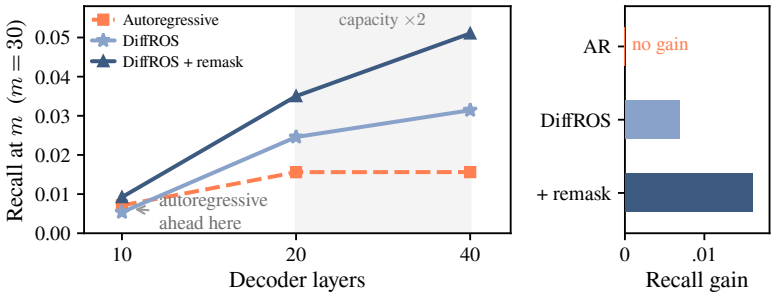

Figure 2: Recall for a fixed m and varied capacity. Right: we show the gain that each decoder takes from the last doubling.

Table 3: Scaling in slate size for both slate factorizations at every m for same decoder size. We read hit rate and recall at cutoff m. Valid is the fraction of emitted identifiers that resolve to an item.

m	Arm	Hit rate	Recall	dist	NFE	valid
10	Autoregressive slate	.2131	.034097	2.13	30	.8243
	DiffROS	.2358	.046339	4.21	3	.8330
	DiffROS + remasking	.2821	.063514	9.26	12	.7947
30	Autoregressive slate	.2569	.014291	2.16	90	.8304
	DiffROS	.3200	.029532	8.32	3	.8205
	DiffROS + remasking	.3892	.047848	21.25	12	.7780
50	Autoregressive slate	.2832	.010348	2.34	150	.8303
	DiffROS	.3822	.025823	13.63	3	.7868
	DiffROS + remasking	.4576	.045383	37.37	12	.7452
100	Autoregressive slate	.3059	.006203	2.32	300	.8401
	DiffROS	.4334	.018366	17.02	3	.7858
	DiffROS + remasking	.5255	.038006	60.09	12	.7260

two and two and a half distinct identifiers, as it does at ten, while paying three hundred forward passes for the hundred.

We analyze decoder compute cost along two relevant dimensions in Figure 3. In sequential depth, measured by the number of dependent decoding steps, DiffROS remains flat as slate size grows, while the autoregressive decoder grows as $3m$. Activated compute grows linearly for every decoder, with DiffROS a constant multiple above. The lower panels show the same tradeoff, where slate size becomes parallel width.

Sequential depth also understates what a duplicate demands of the autoregressive decoder. Once the first levels of an item enter the causal prefix they enter the key value cache, so repairing each duplicated slot takes a full identifier's worth of forward passes. A parallel canvas repairs every duplicated slot together, in one identifier's worth of passes in total. At $m = 30$ the autoregressive decoder produced roughly twenty eight duplicated slots, which is about eighty four forward passes against three. These results characterize computational structure, but not production scale wall clock cost. We report no production scale wall clock measurement. In Appendix F we report denoising depth, repair rounds and the reopening rule trade forward passes against duplicates at decode time.

4.3 Analysis of the Order Aware Metric

We next investigate whether the commonly reported metrics distinguish slate quality from properties of individual predictions. To isolate the scoring rule, we evaluate the same outputs both with the commonly implemented order aware metric and by crediting each target only once. Under the standard implementation, a repeated correct prediction receives credit at every occupied rank. Consequently, three distinct correct items and one correct item repeated three times both receive a perfect nDCG@3.

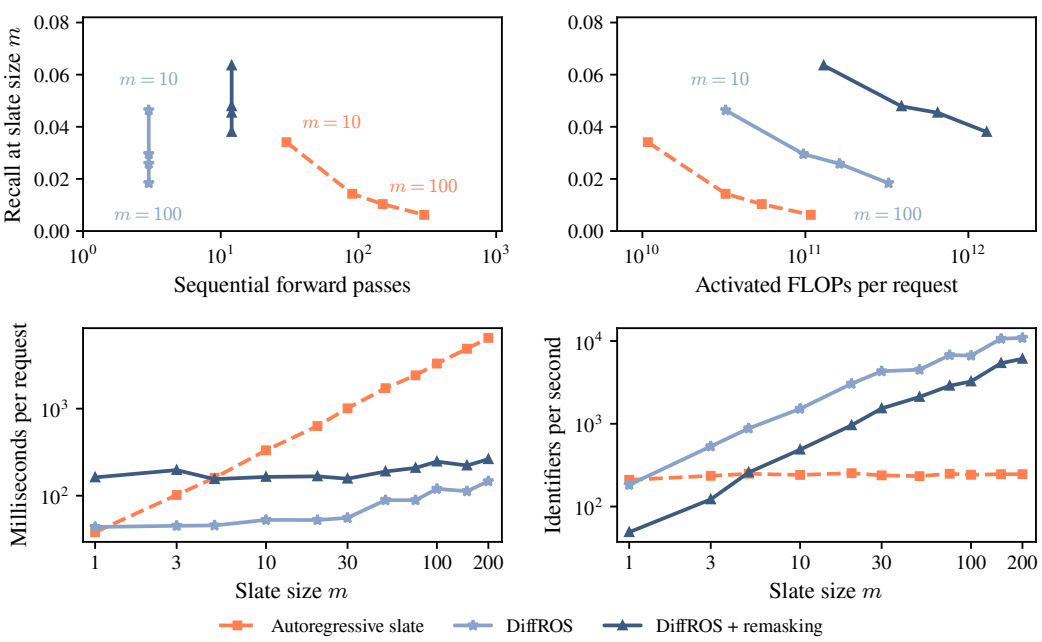

Figure 3: Accuracy against two compute cost axes on the internal corpus (top), and a decode benchmark against slate size (bottom). At the top, we plot each decoder at $m = 10$, 30, 50, and 100, and one decoder per panel carries the endpoint labels.

Table 4: Duplicate accounting in nDCG@30. The duplicate safe $\text{nDCG}^{\dagger}$ credits each target once. Bimodal target, relevance keyed, $m = 30$. Rows are ordered by recall.

Arm	nDCG	rank	$\text{nDCG}^{\dagger}$	rank	Recall	rank	dup
Beam over one item	.14037	3	.14037	1	.085372	1	.0000
DiffROS (r=3)	.06148	4	.05586	2	.029734	2	.3009
DiffROS (r=0)	.14091	2	.04360	4	.017602	3	.7908
Autoregressive slate	.21917	1	.05000	3	.013840	4	.9195

This scoring rule reorders the decoders. The autoregressive slate decoder ranks best despite having the worst recall of the four decoders. Its collapse onto a repeated item is the open catalog counterpart of the duplicate failure mode reported for a directly converted block diffusion decoder (Zhang et al., 2026b). We attribute this result to arithmetic rather than behavior, since it emits about two and a half distinct identifiers across thirty slots, allowing each correct item to receive credit at roughly a dozen ranks, whereas beam emits thirty distinct identifiers and receives credit once per item. Crediting each target once returns beam to the top, as shown in Table 4. Across three independent rounds, the inflation tracks each decoder's duplicate rate.

Neither column, however, reveals how a decoder arranges its slate. Both are modular in the sense of Section 3. They score items separately, cannot reveal an arrangement, and are dominated by hit count. We therefore use the half split from Section 4.1, which conditions on retrieval, so an item that a decoder never recovered enters neither half. This instrument responds only to order, and shuffling DiffROS's own slate reduces its gap almost to zero.

4.4 Ordering Conditioned on Retrieval

Three controls internal to Table 1 rule out the obvious explanations. The effect is not diversity: beam keeps every slot distinct yet orders second weakest, while DiffROS without remasking retains a fifth as many identifiers and seven times the ordering. It is not slate factorization: the autoregressive decoder has slots and slot embeddings yet ranks weakest. And it is not remasking, which barely

changes ordering while tripling diversity. As shown in the last three columns of Table 1, the decoders separate by margins that are large relative to those from the standard metrics. DiffROS separates its halves by several times the beam margin on both targets, and the factorization ordering is identical at all three identifier prefix depths. At an exact identifier match, DiffROS recovers over half of the separation imposed by the bimodal construction, whereas beam recovers about a sixth. Beam still leads at every retrieval cutoff. A second instrument measures a different axis using a measured corpus average as its reference. It separates the decoders in the same direction and by a similar factor (Appendix E).

5 Conclusion

Prior decoders denoise identifier tokens for one next item and assemble a list from beams, while residual quantizer identifiers are ordered by construction, so predicting their levels in parallel fights the tokenizer. Slate slots are exchangeable apart from target order. We transposed the axes on which a diffusion decoder works, so DiffROS keeps the level axis sequential and parallelizes the slot axis, making decoding depth constant in slate size. To the best of our knowledge, it is the first diffusion based decoder to perform generative retrieval of an ordered slate of semantic IDs over an open catalog.

Conditioned on retrieval, DiffROS separates the two halves of its slate several times more sharply than beam search, while the ordering of the three factorizations stays the same at all three identifier prefix depths (Table 1). Its advantage over the autoregressive slate decoder grows monotonically with slate size as was shown in Table 3. At the largest slate size measured, DiffROS recovers close to three times as many targets. The autoregressive slate decoder returns between two and two and a half distinct identifiers whatever slate size it is asked for.

Once visible, this structure makes decoding a control chosen after training. Denoising steps, repair rounds, and slot reopening are decode-time choices from one checkpoint, so the tradeoff between duplicates and forward passes is set at inference. DiffROS also holds depth flat as the slate grows while autoregressive decoding grows linearly; on the internal corpus at a slate of one hundred, it uses three forward passes against three hundred in Figure 3.

Generative retrieval makes the decoder responsible for both finding items and arranging them, and the standard objectives observe only the first task. Expected $\mathrm{recall@}K$ and $\mathrm{nDCG@}K$ are sums of independent item scores, so they discard every relation between two selected items. No term coupling two slots can change the optimum. A next-item decoder with an item-isolated mask estimates exactly this per-item marginal, making beam search over one item aligned with evaluation by construction. It wins on both corpora at equal or lower sequential decoding depth, as Section 3 predicts (Table 1).

The result made us inspect what the order-aware metric counts. It credits a repeated correct prediction at every rank it occupies, so three distinct correct items and one correct item repeated three times both earn a perfect $\mathrm{nDCG@3}$. Crediting each target once returns beam to the top of Table 4, and the inflation tracks each decoder's duplicate rate.

Limitations and future work

DiffROS's training activations are roughly four times the autoregressive decoder's, since it retains all four masked view passes through the denoising loop. Our compute cost claims are bounded from the other side too. Kernel level work achieves an order of magnitude latency reduction at zero quality loss on exactly the query shape an autoregressive slate decoder presents, so a factorization level claim should be discounted by what a compiler already provides (Fang et al., 2026). Counting forward passes cuts both ways, since it also omits the repair compute cost that decoder would use on a duplicated slate.

AI use statement

In this work, we used generative AI tools for polishing the writing, modifying figures, and creating and editing software code. We have reviewed all AI-assisted work. The authors checked the text

and figures against the experimental records in several rounds of review, and verified and tested the code before use.

REFERENCES

Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, 34:17981–17993, 2021.

Vansh Bansal, Cho Cholyeon, Syamantak Kumar, Sujay Sanghavi, and Purnamrita Sarkar. Understanding parallel samplers in masked diffusion via random walks on graphs. *arXiv preprint arXiv:2606.22976*, 2026.

Irwan Bello, Sayali Kulkarni, Sagar Jain, Craig Boutilier, Ed Chi, Elad Eban, Xiyang Luo, Alan Mackey, and Ofer Meshi. Seq2Slate: Re-ranking and slate optimization with RNNs. *arXiv preprint arXiv:1810.02019*, 2018.

Michael Benigni, Maurizio Ferrari Dacrema, and Dietmar Jannach. Diffusion recommender models and the illusion of progress: A concerning study of reproducibility and a conceptual mismatch. *ACM Transactions on Recommender Systems*, 2026.

Zalán Borsos, Matt Sharifi, Damien Vincent, Eugene Kharitonov, Neil Zeghidour, and Marco Tagliasacchi. SoundStorm: Efficient parallel audio generation. *arXiv preprint arXiv:2305.09636*, 2023.

Chen and Karger. Less is more: Probabilistic models for retrieving fewer relevant documents. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 2006.

Romain Deffayet, Thibaut Thonet, Jean-Michel Renders, and Maarten de Rijke. Generative slate recommendation with reinforcement learning. *arXiv preprint arXiv:2301.08632*, 2023.

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

Zhou Fang, Yuhang Huang, Ang Zhang, Yihan He, Ruichao Xiao, Chao Li, Yavuz Yetim, Sibyl Yang, Xiaohan Wei, Fei Tian, Liang Wang, Chonglin Sun, Liyuan Li, Nathan Yan, and Gaoxiang Liu. GRACE: Generative recommender acceleration engine for real-time ads retrieval. *arXiv preprint arXiv:2608.00938*, 2026.

Yufei Feng, Yu Gong, Fei Sun, Junfeng Ge, and Wenwu Ou. Revisit recommender system in the permutation prospective. *arXiv preprint arXiv:2102.12057*, 2021.

Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. Are we really making much progress? a worrying analysis of recent neural recommendation approaches. In *Proceedings of the 13th ACM Conference on Recommender Systems (RecSys)*, 2019.

Francesco Giuliari, Gianluca Scarpellini, Stuart James, Yiming Wang, and Alessio Del Bue. Positional diffusion: Ordering unordered sets with diffusion probabilistic models. *arXiv preprint arXiv:2303.11120*, 2023.

Michael D. Gordon and Peter Lenk. When is the probability ranking principle suboptimal? *Journal of the American Society for Information Science*, 43(1):1–14, 1992.

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

Yupeng Hou, Jiacheng Li, Ashley Shin, Jinsung Jeon, Abhishek Santhanam, Wei Shao, Kaveh Hassani, Ning Yao, and Julian McAuley. Generating long semantic IDs in parallel for recommendation. *arXiv preprint arXiv:2506.05781*, 2025.

Yupeng Hou, Haven Kim, Clark Mingxuan Ju, Eduardo Escoto, Neil Shah, and Julian McAuley. Expressiveness limits of autoregressive semantic ID generation in generative recommendation. *arXiv preprint arXiv:2605.06331*, 2026.

Yang Hu, Jiayi Guo, Jingui Ma, Ning Li, Jiangling Qin, Yanming Li, Yang Deng, Xiaoshuang Chen, and Kaiqiao Zhan. Once generated, ranked: End-to-end generative slate recommendation with unified semantic-collaborative IDs. *arXiv preprint arXiv:2608.17613*, 2026.

Pengyue Jia, Hailan Yang, Shuchang Liu, Xiaobei Wang, Wanyu Wang, Xiang Li, Yongqi Liu, Kaiqiao Zhan, Kun Gai, and Xiangyu Zhao. UniRank: Unified list-wise reranking via confidence-ordered denoising. *arXiv preprint arXiv:2605.10527*, 2026.

Clark Mingxuan Ju, Liam Collins, Leonardo Neves, Bhuvesh Kumar, Louis Yufeng Wang, Tong Zhao, and Neil Shah. Generative recommendation with semantic IDs: A practitioner's handbook. *arXiv preprint arXiv:2507.22224*, 2025.

Nitin Kedia, Saurabh Agarwal, Myungjin Lee, and Aditya Akella. Sangam: Efficiently serving diffusion LLMs with the AR stack. *arXiv preprint arXiv:2607.04206*, 2026.

Jiyeon Kim, Sungik Choi, Yongrae Jo, Moontae Lee, and Minjoon Seo. Early decisions matter: Proximity bias and initial trajectory shaping in non-autoregressive diffusion language models. *arXiv preprint arXiv:2604.10567*, 2026.

Walid Krichene and Steffen Rendle. On sampled metrics for item recommendation. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 1748–1757, 2020.

Hermann Kumbong, Xian Liu, Tsung-Yi Lin, Ming-Yu Liu, Xihui Liu, Ziwei Liu, Daniel Y. Fu, Christopher Ré, and David W. Romero. HMAR: Efficient hierarchical masked auto-regressive image generation. *arXiv preprint arXiv:2506.04421*, 2025.

Ming Li, Sami Jullien, Mozhdeh Ariannezhad, and Maarten de Rijke. A next basket recommendation reality check. *arXiv preprint arXiv:2109.14233*, 2021.

Xinyu Lin, Chaoqun Yang, Wenjie Wang, Yongqi Li, Cunxiao Du, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. Efficient inference for large language model-based generative recommendation. *arXiv preprint arXiv:2410.05165*, 2024.

Xinyu Lin, Haihan Shi, Wenjie Wang, Fuli Feng, Qifan Wang, See-Kiong Ng, and Tat-Seng Chua. Order-agnostic identifier for large language model-based generative recommendation. *arXiv preprint arXiv:2502.10833*, 2025a.

Zhijie Lin, Zhuofeng Li, Chenglei Dai, Wentian Bao, Shuai Lin, Enyun Yu, Haoxiang Zhang, and Liang Zhao. GReF: A unified generative framework for efficient reranking via ordered multi-token prediction. *arXiv preprint arXiv:2510.25220*, 2025b.

Shuchang Liu, Fei Sun, Yingqiang Ge, Changhua Pei, and Yongfeng Zhang. Variation control and evaluation for generative slate recommendations. *arXiv preprint arXiv:2102.13302*, 2021.

Zhao Liu, Yichen Zhu, Yiqing Yang, Xiao Lv, Guoping Tang, Rui Huang, Qiang Luo, Ruiming Tang, Kun Gai, and Guorui Zhou. DiffGRM: Diffusion-based generative recommendation model. In *Proceedings of the ACM Web Conference 2026 (WWW)*, pp. 5853–5864. ACM, 2026.

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.

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.

Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. *arXiv preprint arXiv:2406.03736*, 2024.

Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan H. Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Q. Tran, Jonah Samost, Maciej Kula, Ed H. Chi, and Maheswaran Sathiamoorthy. Recommender systems with generative retrieval. *arXiv preprint arXiv:2305.05065*, 2023.

Houxing Ren, Mingjie Zhan, Zimu Lu, Ke Wang, Yunqiao Yang, Haotian Hou, Junting Pan, and Hongsheng Li. Edit-based refinement for parallel masked diffusion language models. *arXiv preprint arXiv:2605.09603*, 2026.

Yuxin Ren, Qiya Yang, Yichun Wu, Wei Xu, Yalong Wang, and Zhiqiang Zhang. Non-autoregressive generative models for reranking recommendation. *arXiv preprint arXiv:2402.06871*, 2024.

S. E. Robertson. The probability ranking principle in IR. *Journal of Documentation*, 33(4):294–304, 1977. doi: 10.1108/eb026647.

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *arXiv preprint arXiv:2406.07524*, 2024.

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

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K Titsias. Simplified and generalized masked diffusion for discrete data. *arXiv preprint arXiv:2406.04329*, 2024.

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.

Federico Tomasi, Francesco Fabbri, Justin Carter, Elias Kalomiris, Mounia Lalmas, and Zhenwen Dai. Prompt-to-Slate: Diffusion models for prompt-conditioned slate generation. *arXiv preprint arXiv:2408.06883*, 2024.

Anna Volodkevich, Danil Gusak, Anton Klenitskiy, and Alexey Vasilev. Autoregressive generation strategies for top-k sequential recommendations. *arXiv preprint arXiv:2409.17730*, 2024.

Weiqin Yang, Bohao Wang, Zhenxiang Xu, Jiawei Chen, Shengjia Zhang, Jingbang Chen, Canghong Jin, and Can Wang. BEAR: Towards beam-search-aware optimization for recommendation with large language models. *arXiv preprint arXiv:2601.22925*, 2026.

Lin Yao. Decoding in order-agnostic language models: Chain-rule deviation and uniform spreading. *arXiv preprint arXiv:2606.00997*, 2026.

Jaehoon Yoo, Wonjung Kim, Chanhyuk Lee, and Seunghoon Hong. Infinite mask diffusion for few-step distillation. *arXiv preprint arXiv:2605.10518*, 2026.

Qian Zhang, Lech Szymanski, Haibo Zhang, and Jeremiah D. Deng. How reliable are semantic-ID tokenizer comparisons in generative recommendation? In *Proceedings of the 35th ACM International Conference on Information and Knowledge Management (CIKM)*, 2026a.

Zhuoxuan Zhang, Kangqi Ni, Yuhang Chen, Mingfu Liang, Xiaohan Wei, Yunchen Pu, Fei Tian, Chonglin Sun, Frank Shyu, Adam Song, Sandeep Pandey, Luke Simon, Tianlong Chen, and Xi Liu. Diffusion-GR2: Diffusion generative reasoning re-ranker. *arXiv preprint arXiv:2607.01170*, 2026b.

Guorui Zhou, Hengrui Hu, Hongtao Cheng, Huanjie Wang, Jiaxin Deng, Jinghao Zhang, Kuo Cai, Lejian Ren, et al. OneRec-V2 technical report. *arXiv preprint arXiv:2508.20900*, 2025.

A WHAT EACH INSTRUMENT CAN REWARD

Every claim in this paper is made through some instrument, and no instrument in this setting is unqualified. Table 5 collects them with the qualification that travels with each. The grouping is by modularity, which decides whether an instrument can respond to a slate rather than to its items separately. A modular set function is a sum of per item terms, so it cannot reward which items occur together. Modularity therefore decides what an instrument can reward. The upper block scores each item independently, so by the argument of Appendix B it is maximized by ranking items on their marginals and cannot reward structure across slots, while each entry in the lower block is not a sum of per item terms and can in principle respond to arrangement, at the tradeoff stated beside it.

A separate demand arises when emitted identifiers do not resolve to catalog items. Between 73 and 84 percent of emitted identifiers resolve to a real item. This behavior is not an artifact of undertraining, and it is not flat in slate size. The asymmetry between decoders therefore matters for any diversity claim.

Table 5: Instruments and their qualifications. Mod. states whether the instrument is modular.

Instrument	Mod.	What it can see	Qualification
Recall at cutoff K	yes	set overlap; duplicates collapse in it	none; a population mean
Duplicate safe nDCG at K	yes	sorting, not coupling	the common implementation inflates with a decoder's duplicate rate
Hit rate at slate size m	no	at least one correct item per request	levels not comparable across m; cross decoder ratio at fixed m only
Target slot half split	no	ordering among recovered items	duplicate weighted; within decoder contrast and cross decoder ratio only
Kendall's τ; perfect match ratio	no	rank agreement on exactly recovered items	basis proportional to recall, so not comparable across decoders
α-nDCG; subtopic recall at K	no	diversity among retrieved items	sums over retrieved items, so a retrieval deficit dominates
Valid emission rate; distinct identifiers per slate	n/a	whether an emission resolves to an item	50k sample, $\pm 0.4\%$, decoder asymmetric; distinct counts identifiers, not posts

B THE FULL MODULARITY ARGUMENT

In Section 3, we state the property and its consequence, and here we give the algebra behind them. This is the finite cutoff form of a known result (Robertson, 1977; Chen & Karger, 2006; Gordon & Lenk, 1992).

We fix a request and write V for the catalog and $G \subseteq V$ for the request's ground truth target set. The set G is random, satisfies $|G| \geq 1$, and may have any joint law. At cutoff K, we score the set $S \subseteq V$ of the first K distinct items from the decoder, using $\mathrm{recall@}K(S) = |S \cap G|/|G|$. Linearity of expectation then gives

$$\mathbb{E}[\mathrm{recall@}K(S)] \;=\; \sum_{i \in S} w_i, \qquad w_i \;=\; \mathbb{E}[\mathbf{1}\{i \in G\} \,/\, |G|]\,.$$

The expected objective is therefore modular in S, since its value is the sum of a separate weight for every item in S. Maximizing this modular function under $|S| \leq K$ is equivalent to maximizing it over a uniform matroid, and the solution contains the K largest w_i. The univariate marginal w_i integrates all dependence among the elements of G into a separate weight for each item. This dependence is therefore invisible to both the objective and its maximizer.

Sorting by rank does not alter this conclusion. We let $d_j = 1/\log_2(j+1)$ denote the discount at rank j, and we let $v_i = \mathbb{E}[\, g(i;G)/Z(G)\,]$. Here the gain g and ideal DCG Z depend only on G. For ordered tuples π of K distinct items,

$$\mathbb{E}[\mathrm{nDCG@}K(\pi)] \;=\; \sum_{j=1}^{K} d_j \, v_{\pi(j)},$$

and the rearrangement inequality assigns the K largest v_i in decreasing order. The same per item marginal therefore determines both the set and its ordering. A rank discount makes the metric sensitive to sorting, but not to which items are selected together.

We can avoid the argument through a utility that is not a sum of per item terms, or through a constraint that is not a uniform matroid. This is the axis on which the argument above does not apply to a joint decoder.

C THE MASKED DIFFUSION CONSTRUCTION ON A TWO AXIS CANVAS

Section 2 states the forward process and the bound. This appendix specifies the reverse process and the two departures of our training objective from the bound. Appendix D then turns to what each factorization estimates. This is the standard masked diffusion construction (Austin et al., 2021; Shi et al., 2024; Sahoo et al., 2024; Ou et al., 2024; Nie et al., 2025) applied to a two axis canvas.

FORWARD AND REVERSE PROCESS

Let u denote the encoded request and $x_0 \in \mathcal{V}^{m\times C}$ the clean canvas, with $x_0^{j\ell}$ denoting the token at slot j and identifier level ℓ. Equation (1) defines the forward process, in which each position is independently replaced by M with probability $\pi_\ell(t)$. Thus, x_1 is fully masked and x_0 is fully observed. The reverse process also factorizes. For $0 \le s < t \le 1$,

$$q_{s|t}\big(x_s^{j\ell} \mid x_t\big) = \begin{cases} 1, & x_t^{j\ell} \neq \mathrm{M},\ x_s^{j\ell} = x_t^{j\ell}, \\ \dfrac{\pi_\ell(s)}{\pi_\ell(t)}, & x_t^{j\ell} = \mathrm{M},\ x_s^{j\ell} = \mathrm{M}, \\ \left(1 - \dfrac{\pi_\ell(s)}{\pi_\ell(t)}\right) q_{0|t}\big(x_s^{j\ell} \mid x_t\big), & x_t^{j\ell} = \mathrm{M},\ x_s^{j\ell} \neq \mathrm{M}, \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

A resolved position remains fixed. A masked position remains masked with probability $\pi_\ell(s)/\pi_\ell(t)$ and is otherwise sampled from the data prediction $q_{0|t}$. Setting $\pi_\ell(t) = t$ gives the isotropic transition, for which the surviving mask fraction is s/t (Nie et al., 2025). Under the level ordered schedule, the ratio is one at every level except the level currently being resolved. The reverse process therefore commits one level at a time across all slots.

THE DATA PREDICTION IS TIME FREE

The network needs to estimate only $q_{0|t}$. At every masked position,

$$q_{0|t}\big(x^{j\ell} \mid x_t\big) \;=\; p_{\mathrm{data}}\Big(x_0^{j\ell} \mid u,\, x_t^{\mathrm{UM}}\Big), \tag{6}$$

where x_t^{UM} denotes the unmasked entries of x_t (Ou et al., 2024). By construction, these entries retain their values from x_0. The conditional thus depends on t only through the set of unmasked positions, not through t itself. This argument does not require a shared rate across positions and therefore applies unchanged to a per level schedule. Two practical consequences follow. The mask predictor requires no time input. Moreover, the number of denoising steps is a decode time choice that requires no retraining. Consequently, T can be swept for a fixed checkpoint, whereas m cannot, because changing m changes which slot embeddings receive gradient.

THE BOUND AND THE DEPARTURES OF OUR TRAINING OBJECTIVE

For the isotropic schedule $\pi_\ell(t) = t$, the masked cross entropy in Equation (2) upper bounds the negative log likelihood of the slate conditioned on the request:

$$-\,\mathbb{E}_{(u,x_0)}[\log p_\theta(x_0 \mid u)] \;\le\; \mathcal{L}(\theta), \tag{7}$$

which is the conditional masked diffusion bound (Shi et al., 2024; Sahoo et al., 2024; Ou et al., 2024; Nie et al., 2025). Our training objective departs from this bound in two ways.

First, we do not sample training masks from Equation (1). Training uses the on policy scheme of Equation (3) (Liu et al., 2026), in which the model selects the masked set, so the induced mask distribution depends on θ. Equation (7) requires a forward process fixed independently of the parameters. We therefore do not claim that it bounds the objective we optimize. We adopt this scheme from a decoder paired with a tokenizer whose levels are independent, whereas ours are residual.

Second, the loss averages over a small number of nested views instead of a single sample of t. This departure can be reduced in two ways, and one changes the objective. Averaging internally normalized views exactly recovers the $1/t$ weighting of Equation (2), because the induced weight of each position is inversely proportional to the number of masked positions in its view. Our default on the internal corpus instead weights every supervised position equally. This choice flattens the weighting and therefore departs from the bound. We retain it because it measured better on that corpus. The resulting objective is no longer the likelihood bound in Equation (7).

D What each factorization estimates

Appendix C gives the construction. This appendix draws its consequences: what slot exchangeability implies for the position wise loss, what quantity each of the three attention masks asks the model to estimate, and why remasking is a selector choice rather than a transition of the reverse process.

Slot exchangeability makes the position wise loss one term of a larger objective

The canvas axes differ in both structure and symmetry. Identifier levels are residual and admit no permutation. When the target is a set rather than a list, slots admit $m!$ permutations. If the target is the set S with $|S| = m$, the relevant quantity for a set valued task is

$$p_\theta(S \mid u) \;=\; \sum_{\sigma \in \mathfrak{S}_m} p_\theta(x_0^\sigma \mid u)\,, \tag{8}$$

where x_0^σ assigns the elements of S to slots according to σ. The position wise cross entropy used for training retains only the term $\sigma = \mathrm{id}$. It therefore upper bounds the set negative log likelihood, with gap $\log\big(\sum_\sigma p_\theta(x_0^\sigma \mid u) / p_\theta(x_0^{\mathrm{id}} \mid u)\big)$, which is at most $\log m!$ when the identity assignment is the most likely one. Replacing the sum with its largest term yields the matching objective optimized by set prediction architectures.

This formulation yields a testable consequence. The gap is small exactly when one assignment dominates the sum. For a target with substantive order, as in the bimodal construction of Section 4.1, the identity assignment is correct by construction, so it dominates and the gap closes. For a target formed as a ranked list by a pointwise scorer, adjacent ranks induce nearly identical conditionals, the terms in Equation (8) nearly coincide, and the gap approaches its $\log m!$ ceiling. That ceiling reflects the interchangeability of the slots rather than structure a matching objective could recover: when every assignment is equally good, matching has nothing to choose between. Our measurements include both regimes. The formulation explains why one regime was not expected to reward a matching loss.

What each factorization estimates

The three masks in Section 2 do not define three decoders for the same distribution. They estimate different objects. This distinction explains why a next item decoder read out with beam search is a strong baseline.

Consider the item isolated mask $I_m \otimes \mathrm{tril}(C)$, trained without a slot embedding or positional term. Position $(j, 1)$ attends only to itself and receives a token that is identical for every j. Its logits therefore do not depend on j. The same property holds at every level, conditioned on the prefix. The loss for each request becomes

$$\sum_{j=1}^{m} \sum_{\ell=1}^{C} \mathrm{CE}\Big(f_\ell\Big(u, c^j_{<\ell}\Big)\,,\, c^j_\ell\Big)\,, \tag{9}$$

a sum of m identical terms evaluated on m distinct targets. Its minimizer is the empirical distribution over the m target items, factorized along the identifier. This decoder therefore performs maximum likelihood estimation of one categorical $p(\text{item} \mid u)$ using m samples per request. A beam of width m returns its top m modes, subject to the pruning loss noted in Section 3. Section 3 further shows that the reported metrics reward top K under precisely this marginal. The baseline is therefore aligned with the evaluation by construction.

Beam search also requires C forward passes for any width and slate size. Thus, sequential depth mC arises from the slate factorization rather than from autoregression itself (Hu et al., 2026). Counting forward passes therefore favors beam. Comparison with an autoregressive decoder that has not been kernel profiled or given a lossless acceleration pass also favors beam (Fang et al., 2026; Lin et al., 2024; Kedia et al., 2026).

The autoregressive slate mask $\mathrm{tril}(mC)$ estimates a different object, m rank conditionals $p(\text{item at slot } j \mid u, \text{slots } < j)$, each from one sample. Selecting one argmax per slot recovers a slate only when these conditionals differ. If they nearly coincide, and the catalog remains open so that no mechanism masks an item already emitted by the decoder, their argmaxes also coincide. This is the absorbing state reported in Section 4. Under this formulation, it follows from the quantity that the mask asks the model to estimate, rather than from a failure of the decoder.

Remasking as a selector choice

A reverse step that reopens committed positions is specified by the set of positions it reopens. Let $\hat{x}$ denote the current prediction and c its per position confidence. Three selectors appear in the literature and in our decoders. Uniform selection among committed positions is the only selector consistent with Equation (5), and it defines the model distribution. Lowest confidence selection, $R = \arg\min_c$, is the standard heuristic. Our selector instead reopens the redundant slot:

$$R \;=\; \left\{ j \;:\; \exists\, j' \neq j,\; \hat{x}^j = \hat{x}^{j'},\; c^j < c^{j'} \right\}, \tag{10}$$

reopened at every level of the slot, with the emitted identifier excluded from the reprediction. This mechanism reflects the domain rather than the decoder. In text, a repeated token is genuinely improbable, so confidence incidentally detects redundancy and the second selector suffices. On a slate, the repeated item is the mode and thus occupies the highest confidence slot. A confidence keyed selector consequently reopens the positions least in need of repair while preserving the duplicates. The last two selectors condition on the realized sample rather than on t. Neither is a transition of Equation (5); both are decode time repair operators, and we report them accordingly.

E A second ordering axis with a measured reference

For the half split from Section 4.1, we can use the teacher's relevance score instead of the target's index. On the relevance keyed target, we measure the mean relevance of the items each decoder recovered and split them at that decoder's own midpoint. The favorable direction reverses. A higher first half is the good direction here, which is the opposite of the target slot instrument, and a well ordered decoder places its higher relevance recoveries in its first half.

Under this self split, DiffROS separates its halves by $+0.273$, compared with $+0.049$ for beam. This gives a $5.5\times$ within decoder separation (Table 6). DiffROS's second half is .4990, essentially equal to the corpus average, whereas beam's second half is .6597, well above that reference. We constructed the instruments at different times, and they operate along different axes and use different references. One uses a measured shuffle null, and the other uses a measured corpus average. Their agreement in direction and their similar separation factors provide the main robustness argument available for Section 4.4.

We interpret only the within decoder split, where each decoder's selection bias cancels, because absolute levels across decoders remain confounded. DiffROS recovers 20.97 distinct identifiers of thirty, compared with beam's 30.00. The averages therefore cover sets with different sizes and compositions. A decoder that recovers only identifiers for which it is confident may, by construction, select an easier subsample. DiffROS also has higher absolute relevance than beam in the first half, with .7722 against .7091. We see this difference in the measurements, but it is precisely the comparison that the confound prevents us from emphasizing.

Table 6: The relevance axis order instrument at $m = 30$.

Arm	first half	second half	gap	ratio
Beam over one item	.7091	.6597	$+0.049$	$1.07\times$
DiffROS (r=3)	.7722	.4990	$+0.273$	$1.55\times$

F DECODE TIME ABLATIONS

Denoising depth, repair rounds and the choice of which slot to reopen are all set at decode time, so they can be studied on a single trained checkpoint. This appendix reports that study.

We investigate how denoising depth, repair rounds, and the choice of reopened slots affect inference when model fitting is held fixed. All results in this section use one trained checkpoint, with the same weights for denoising, repair, and decisions about which slots to reopen. On Amazon repair improves recall on all three categories, though by far less than on the internal corpus.

We find that which slots are reopened matters as much as the number of repair rounds. At a matched budget of six forward passes, reopening the redundant slot yields $.00922$, whereas reopening the least confident slot yields $.00767$, despite identical weights. Within a slate, the repeated item is the mode and therefore has the highest confidence. A confidence keyed selector instead reopens items that are unique but uncertain, leaving duplicates unchanged.

Relative to decoding without repair, repair improves recall at every slate size in Table 3, from 37 percent at $m = 10$ to 107 percent at $m = 100$.

Additional rounds exhibit the same pattern, with the effect saturating early. Three rounds attain 91 percent of the recall achieved by ten rounds at 36 percent of the compute cost, requiring twelve forward passes rather than thirty three. Beyond this point, each additional round primarily removes duplicates that no longer contribute recall.