RETHINKING TOKENIZATION FOR DISCRETE DIFFUSION LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Discrete diffusion models offer a potential speed advantage over autoregressive models by generating multiple tokens per denoising step. However, parallel decoding samples each position independently within a denoising step. This ignores the dependence between tokens generated in the same step. This discrepancy, known as factorization error, grows as more tokens are decoded in parallel. Prior work has sought to reduce this error through sampling schedules or auxiliary models, but these approaches address which tokens are generated together without changing the tokens themselves. We instead investigate tokenization itself as a design choice that shapes this error. Discrete diffusion models commonly inherit tokenizers designed for autoregressive models, despite token boundaries determining which dependencies remain between independently predicted outputs. Lossless tokenizers sharing vocabulary size and per-example encoded lengths achieve the same oracle denoising loss, yet they can incur different factorization errors under parallel sampling. We decompose this error exactly by interaction order, separating schedule-determined weights from tokenizer-determined interactions. This decomposition clarifies when pairwise dependence suffices for comparing tokenizers and when higher-order interactions alter the ranking. We compare BPE, WordPiece, Unigram, and a weighted pointwise mutual information (WPMI) merge criterion motivated by the adjacent pairwise component in this decomposition. Experiments on LM1B and OpenWebText show that tokenizers with similar compression and prediction loss can diverge in generation quality as the number of denoising steps decreases. Our findings suggest that tokenizer selection, commonly inherited from autoregressive language-modeling pipelines, may need to be rethought for discrete diffusion language models.

1 INTRODUCTION

Discrete diffusion models have emerged as a promising paradigm for accelerating text generation through parallel token prediction. Unlike autoregressive models that generate one token at a time, they predict multiple tokens in parallel within each denoising step and can thereby reduce the number of sequential decoding steps and lower generation latency. Recent work has extended this approach to large-scale language models (Team et al., 2026; Song et al., 2025; Labs et al., 2025).

However, parallel decoding exposes a gap between training and generation because learning accurate token-wise conditionals does not guarantee a correct joint distribution under independent sampling. Given observed tokens Z_C and update positions B, factorized decoding approximates the joint conditional as $p(Z_B \mid Z_C) \approx \prod_{i \in B} p_\theta(Z_i \mid Z_C)$. Each token is sampled from the same context without conditioning on the other outputs generated in that step. Any remaining dependence among these outputs is therefore discarded, introducing factorization error even when the individual marginals are exact. Prior work has addressed this factorization error by introducing additional models (Liu et al., 2025; Xu et al., 2025) for joint dependence or by selecting denoising schedules (Park et al., 2025) that reduce the dependencies exposed at each step.

These approaches adjust the dependencies exposed during sampling but do not change the tokens themselves. However, tokenization reshapes the dependency structure among tokens by determining which dependencies are absorbed within a single token and which remain between tokens sampled independently. Changing tokenization can therefore change the factorization error even under the

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.

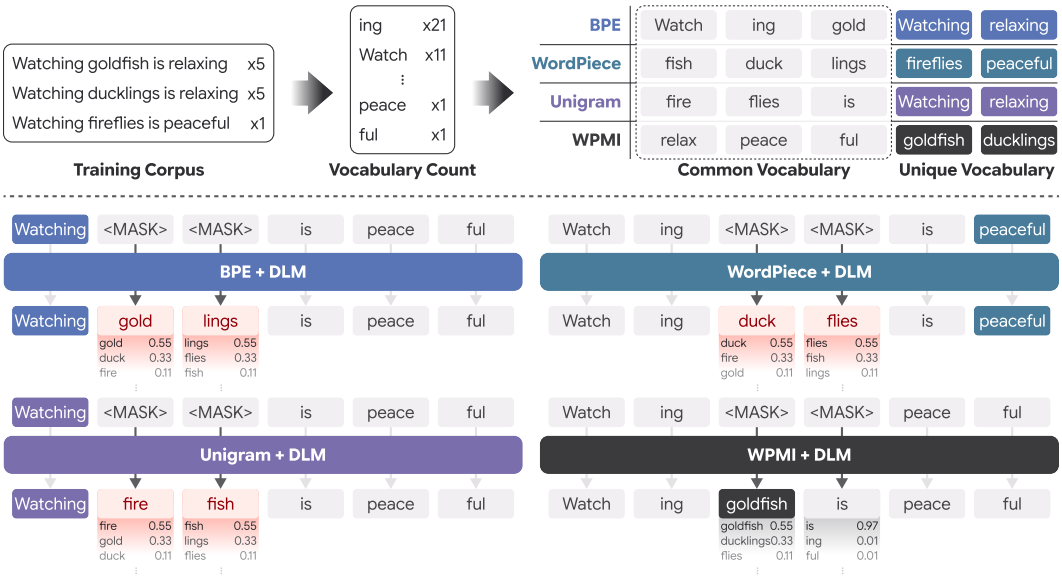


Figure 1: **Tokenization shapes parallel factorization error. Top:** Starting from the same corpus and initial pieces, BPE, WordPiece, Unigram, and WPMI construct different vocabularies under the same budget. **Bottom:** These token boundaries determine which dependencies are modeled within a token versus across independently sampled positions. Fragmented units can produce incompatible combinations (e.g., `gold`+`lings`, `fire`+`fish`), while merging them into a single token avoids this cross-token factorization.

same model architecture and sampling procedure. Despite this influence, discrete diffusion language models commonly retain tokenizers developed for autoregressive language modeling. These tokenizers were designed to optimize corpus statistics and compression efficiency. How tokenizer selection shapes this dependence and the resulting factorization error remains comparatively underexplored.

In this work, we investigate how tokenizer design shapes factorization error in masked diffusion language models. Our contributions are as follows:

- **Tokenization-dependent factorization error.** We show that lossless tokenizers with matched vocabulary size and per-example encoded lengths share the oracle denoising optimum but can incur different parallel sampling errors. Under fixed-size unmasking, we derive an exact decomposition that separates sampling weights from tokenizer-induced interactions.
- **Implications for tokenizer design.** We characterize when pairwise comparisons remain informative despite higher-order interactions and quantify their sensitivity to differences in conditional dependence. Restricting the pairwise term to adjacent pairs actionable by a single merge and decomposing by pair type naturally yields weighted pointwise mutual information (WPMI) as the corresponding merge criterion. While WPMI itself has been used in other contexts such as phrase discovery (Hazen & Richardson, 2012), we motivate it here as a lexical criterion through the analysis of parallel factorization error and study its behavior in masked diffusion language models.
- **Exploration of the tokenization design space.** We compare BPE (Sennrich et al., 2016), WordPiece (Schuster & Nakajima, 2012), Unigram (Kudo, 2018), and a WPMI merge criterion motivated by the adjacent pairwise term, on LM1B (Chelba et al., 2013) and OpenWebText (Gokaslan et al., 2019). We verify that these tokenizers achieve similar compression and held-out prediction quality, and examine whether they remain equally robust as the number of denoising steps decreases.

Together, these analyses establish that parallel factorization error depends not only on how tokens are generated but also on how they are defined.

2 PRELIMINARIES

2.1 TOKENIZATION

Notation. A tokenizer represents a raw string as a sequence of discrete tokens. Let $X \sim p_X$ have finite support in Σ^* for a finite raw alphabet Σ. A deterministic tokenizer $\tau = (V_\tau, E_\tau, D_\tau)$ consists of a finite vocabulary V_τ, an encoder $E_\tau : \Sigma^* \to V_\tau^*$ and a decoder $D_\tau : V_\tau^* \to \Sigma^*$. Its output is $Z^\tau = E_\tau(X)$ with encoded length $N_\tau = |Z^\tau|$. We assume $D_\tau(E_\tau(x)) = x$ on the source support. Write p_τ^Z and p_τ^N for the token-sequence and length laws, and $p_{\tau,n}$ for the token law conditional on $N_\tau = n$. The operators $\mathrm{H}_{\tau,n}$ and $I_{\tau,n}$ use this conditional law. For a fixed tokenizer we write $Z = Z^\tau$. Observed tokens retain their position indices. All logarithms are natural unless stated otherwise.

Standard tokenizers. Let $c(u, v)$ and $c(u)$ denote adjacent-pair and single-token counts in the current corpus. **BPE** (Sennrich et al., 2016; Gage, 1994) constructs its vocabulary by repeatedly merging the pair with the largest $c(u, v)$. For **WordPiece** (Schuster & Nakajima, 2012), we use the association-based merge score $c(u, v)/(c(u)c(v))$. **Unigram** (Kudo, 2018) fits token probabilities $\vartheta(v)$ using likelihood summed over valid segmentations and prunes an overcomplete vocabulary according to the estimated likelihood loss. Encoding selects the segmentation maximizing $\prod_i \vartheta(z_i)$, with a fixed tie-breaking rule. The fitted probabilities ϑ define the segmentation model and are distinct from the induced data distribution p_τ^Z.

2.2 MASKED DIFFUSION MODELS

Masked diffusion corrupts tokens to an absorbing mask state and learns to reverse this corruption (Austin et al., 2021; Sahoo et al., 2024; Shi et al., 2024). At a supported length n, let $Z \sim p_{\tau,n}$, $[n] = \{1, \dots, n\}$, and $\mathsf{m} \notin V_\tau$ be the mask symbol. The noisy state is $Z_t \in (V_\tau \cup \{\mathsf{m}\})^n$, with $\mathcal{M}_t = \{i : Z_{t,i} = \mathsf{m}\}$.

Forward masking. The forward process independently replaces tokens with an absorbing mask m. Once masked, a token remains masked. Starting from $Z_0 = Z$, the time-t marginal factorizes as $q_t(z_t \mid z) = \prod_{i=1}^n q_t(z_{t,i} \mid z_i)$, where

$$q_t(y \mid v) = \alpha_t \mathbf{1}\{y = v\} + (1 - \alpha_t)\mathbf{1}\{y = \mathsf{m}\}. \tag{1}$$

The noise schedule α_t is absolutely continuous and differentiable on $(0, 1)$. It decreases strictly from $\alpha_0 = 1$ to $\alpha_1 = 0$, yielding $Z_1 = \mathsf{m}^n$. Numerical cutoffs are distinguished from these full-noise endpoints.

Denoising objective. A denoiser $p_{\theta,i}(\cdot \mid Z_t, t)$ predicts the clean token at each masked position. Training minimizes the continuous-time negative evidence lower bound (NELBO) conditional on encoded length (Sahoo et al., 2024; Shi et al., 2024):

$$\mathcal{L}_{\mathrm{MDM}}(\theta; \tau) = \int_0^1 \frac{-\dot{\alpha}_t}{1 - \alpha_t} \mathbb{E}_{Z, Z_t}\left[- \sum_{i \in \mathcal{M}_t} \log p_{\theta,i}(Z_i \mid Z_t, t) \right] dt. \tag{2}$$

Here $Z \sim p_\tau^Z$ and $Z_t \sim q_t(\cdot \mid Z)$. The loss sums over masked positions and averages over source sequences without per-token normalization. The length-n setup applies separately to each sequence in this expectation.

Parallel unmasking. Starting from m^n, sampling follows an ordered partition $\mathcal{S} = (B_1, \dots, B_K)$ of $[n]$. Round k independently samples positions in B_k given the previously revealed tokens at $O_k = \bigcup_{\ell<k} B_\ell$. We use a common $\Pi = (\Pi_n)$ across tokenizers, where $\mathcal{S} \sim \Pi_n$ is independent of token values conditional on length n. With exact conditionals, the sampling law is

$$p_{\tau,\mathcal{S}}^{\parallel}(z \mid n) = \prod_{k=1}^K \prod_{i \in B_k} p_{\tau,n}(z_i \mid z_{O_k}). \tag{3}$$

The learned sampler replaces these conditionals with denoiser predictions on the corresponding masked input, with query times determined by n and $\mathcal{S}$ (Appendix A.1).

Factorization error. Independent parallel sampling can introduce factorization error by ignoring dependencies among co-generated tokens, even with exact marginals (Liu et al., 2025). To isolate parallel factorization from prediction error, define

$$\mathcal{E}_{\Pi}(\tau \mid n) := \mathbb{E}_{\mathcal{S}\sim\Pi_n} D_{\mathrm{KL}}\left(p_{\tau,n} \,\middle\|\, p^{\parallel}_{\tau,\mathcal{S}}(\cdot \mid n)\right). \tag{4}$$

Its length average is $\mathcal{E}_{\Pi}(\tau) := \mathbb{E}_{n\sim p^N_\tau}[\mathcal{E}_{\Pi}(\tau \mid n)]$. The KL term measures the excess log loss of the schedule-induced sampling law relative to the true length-conditioned joint, yielding the dependence decomposition in Section 3.2.

3 Why Tokenizers Become Problematic in Discrete Diffusion

Tokenization does not change the total information a denoiser must recover: for any lossless tokenizer, the oracle denoising loss and the encoded-length entropy together equal the source entropy (Section 3.1). However, when multiple tokens are sampled independently in a single round, the resulting factorization error depends on how the tokenizer distributes source dependence across token variables (Section 3.2). This section separates these two effects to isolate the tokenizer-dependent component that persists even with an optimal denoiser.

3.1 Oracle Training Loss under Tokenization

We separate factorization error from the oracle denoising objective to distinguish sampling effects from differences in reconstruction information. Under independent masking, the optimal denoiser at each masked position is the clean-data conditional given the visible tokens. Integrating these conditional entropies over the full noise interval shows that the oracle loss and encoded-length entropy together recover the same source entropy for every lossless tokenizer.

Proposition 1 (Oracle denoising loss)**.** *For the full-noise, sequence-summed objective in equation 2, every deterministic lossless tokenizer has* $\mathcal{L}^*_{\mathrm{MDM}}(\tau) := \inf_\theta \mathcal{L}_{\mathrm{MDM}}(\theta;\tau)$ *satisfying*

$$\mathcal{L}^*_{\mathrm{MDM}}(\tau) = \sum_n p^N_\tau(n)\, \mathrm{H}_{\tau,n}(Z) = \mathrm{H}(X) - \mathrm{H}(N_\tau), \tag{5}$$

where the infimum is over unrestricted conditional predictors.

Appendix A.2 provides the derivation. When the encoded length is constant almost surely, the oracle optimum equals $\mathrm{H}(X)$. More generally, the denoising loss together with the sequence-length entropy $\mathrm{H}(N_\tau)$ recovers the same source entropy. Tokenization therefore redistributes uncertainty across token predictions without changing the total information to be recovered. This isolates the tokenizer-dependent effect studied next, which arises from the factorization imposed by parallel sampling rather than from the oracle denoising objective itself.

3.2 Tokenization-Dependent Factorization Error

We now show that the oracle factorization error $\mathcal{E}_{\Pi}(\tau)$ depends on the tokenizer, not only on the unmasking schedule. Existing analyses fix the token distribution and study schedule design (Lavenant & Zanella, 2025; Chen et al., 2025). We instead fix the raw source p_X and sampling schedule Π, and vary the tokenizer. The following proposition expresses this error as the expected sum of conditional total correlations within sampling rounds.

Proposition 2 (Tokenization-dependent sampling cost)**.** *For a schedule law* Π_n *independent of token values conditional on* $N_\tau = n$*,*

$$\mathcal{E}_{\Pi}(\tau) = \mathbb{E}_{n,\mathcal{S}} \sum_{k=1}^{K} \mathrm{TC}_{\tau,n}(Z_{B_k} \mid Z_{O_k}), \tag{6}$$

where $\mathrm{TC}_{\tau,n}(Z_B \mid Z_O) = \sum_{i\in B} \mathrm{H}_{\tau,n}(Z_i \mid Z_O) - \mathrm{H}_{\tau,n}(Z_B \mid Z_O)$.

Appendix A.3 provides the derivation and learned-marginal error decomposition. Under singleton unmasking such as left-to-right autoregressive generation, each token conditions on all previously

generated values and the factorization error is identically zero regardless of tokenization. Parallel rounds can break this invariance: the error equals the expected sum of within-round conditional total correlations, which depend on how the tokenizer distributes source dependence across token variables. Different tokenizers can therefore induce different $\mathcal{E}_\Pi(\tau)$ under the same schedule law.

Defining the schedule-matched prediction cost $\mathcal{J}_\Pi(\tau) := \mathrm{H}(N_\tau) + \mathbb{E}_{n,\mathcal{S}} \sum_k \sum_{i \in B_k} \mathrm{H}_{\tau,n}(Z_i \mid Z_{O_k})$ gives

$$\mathcal{J}_\Pi(\tau) = \mathrm{H}(X) + \mathcal{E}_\Pi(\tau). \tag{7}$$

Since $\mathrm{H}(X)$ is fixed, minimizing prediction cost over tokenizers is equivalent to minimizing factorization error. We next decompose this error by interaction order to separate the roles of the tokenizer and the schedule.

4 How Tokenization Shapes Factorization Error

Section 3 separates denoising information from parallel factorization error. We now express the error through token interactions and examine when its pairwise component supports tokenizer comparison. We compare lossless tokenizers $\tau \in \mathfrak{T}$ with the same vocabulary size and $N_\tau(x) = N(x)$ on every source sequence. We condition on a supported $N = n \geq 2$ and suppress this conditioning below. Appendix B.1 specifies the resulting common source and sampling geometry.

4.1 An Exact Decomposition at Matched Length

Tokenization shapes both pairwise dependence and interactions among larger groups of tokens. We decompose the factorization error by interaction order to separate these tokenizer-induced interactions from weights determined by the sampling policy.

We consider Π_b, which unmasks b tokens per step, with a possibly shorter final round. At matched length, all tokenizers share the same round sizes. For $2 \leq r \leq n$, define the aggregate order-r interaction $S_r(\tau) := \sum_{A \subseteq [n]:\, |A|=r} I_r(Z^\tau_A)$, where $I_r(Z^\tau_A) := \sum_{\varnothing \neq D \subseteq A} (-1)^{|D|+1} \mathrm{H}(Z^\tau_D)$ is the order-r signed co-information. In particular, $I_2(Z^\tau_i, Z^\tau_j) = I(Z^\tau_i; Z^\tau_j)$.

Theorem 3 (Matched-length interaction decomposition)**.** *For every $\tau \in \mathfrak{T}$,*

$$\mathcal{E}_{\Pi_b}(\tau \mid n) = \sum_{r=2}^{n} (-1)^r \rho_r(n,b) S_r(\tau), \tag{8}$$

where $\rho_r(n,b) = 1 - \binom{n}{r}^{-1} \sum_k |B_k| \binom{|O_k|}{r-1}$ is the probability that the latest occupied round of r specified positions contains at least two of them.

Appendix B.2 provides the proof.

The decomposition separates two complementary design choices: the sampling policy determines how interactions of different orders are weighted, while the tokenizer determines the interactions themselves. At matched length, the weights $\rho_r(n,b)$ are common across tokenizers, so differences in factorization error arise entirely from $S_r(\tau)$. The leading term $\rho_2(n,b)S_2(\tau)$ depends only on pairwise marginals, but higher-order terms can reinforce or offset it. We next examine when pairwise statistics nevertheless suffice for ranking tokenizers.

4.2 What Pairwise Criteria Capture

Merge-based tokenizers are constructed through successive merges of adjacent token pairs, directly controlling which dependencies are absorbed within a token and which remain between separately predicted outputs. The second-order component $\rho_2(n,b)S_2(\tau)$ aggregates mutual information over all position pairs and connects naturally to this process. We characterize when ranking tokenizers by S_2 preserves their ordering by the full factorization error.

Proposition 4 (Stability of pairwise tokenizer comparisons)**.** *Let τ_0, τ_1 have a common encoded length and positive S_2. For $S_2(\tau) > 0$ and $0 \leq s \leq n-2$, define the normalized conditional*

response $g_\tau(s) = \mathbb{E}_{i,j,O:|O|=s} I(Z_i^\tau; Z_j^\tau \mid Z_O^\tau) / \mathbb{E}_{i,j} I(Z_i^\tau; Z_j^\tau)$*, where targets* i, j *are distinct and uniform and* O *is a uniform* s*-subset excluding both. Then for* $b \geq 2$*,*

$$\frac{\mathcal{E}_{\Pi_b}(\tau_1 \mid n)}{\mathcal{E}_{\Pi_b}(\tau_0 \mid n)} \leq \frac{S_2(\tau_1)}{S_2(\tau_0)} \big[1 + \epsilon_b(\tau_1, \tau_0)\big], \tag{9}$$

where $\epsilon_b(\tau_1, \tau_0)$ *weights* $[g_{\tau_1}(s) - g_{\tau_0}(s)]_+$ *by the sampling schedule and normalizes by the reference response Equation 29. The ranking* $S_2(\tau_1) < S_2(\tau_0)$ *implies* $\mathcal{E}_{\Pi_b}(\tau_1 \mid n) < \mathcal{E}_{\Pi_b}(\tau_0 \mid n)$ *whenever* $S_2(\tau_0)/S_2(\tau_1) - 1 > \epsilon_b(\tau_1, \tau_0)$*.*

Appendix B.3 provides the proof. The ranking is preserved whenever the pairwise gap exceeds the response discrepancy; identical responses give proportional errors even with large higher-order corrections. We next use adjacent pair statistics as a practical construction signal without assuming that the discrepancy is negligible.

5 A Lexical Criterion from Pairwise Dependence

Section 4 shows that pairwise dependence can preserve tokenizer comparisons under the conditions characterized there. Turning this observation into a tokenizer, however, requires a local construction rule. Standard merge-based tokenizers build their vocabularies incrementally by combining adjacent token pairs, rather than optimizing the final tokenization globally. We therefore use the pairwise analysis to define a merge criterion: at each step, we score adjacent token pairs by how strongly their values are statistically coupled and merge the highest-scoring pair.

5.1 WPMI Tokenizer Construction

From pairwise dependence to a merge rule. Whereas S_2 aggregates dependence over all position pairs, a lexical merge can act only on neighboring tokens. We therefore restrict attention to adjacent pairs and assign a single score to each pair type (u, v) by aggregating its occurrences across positions and examples. This converts the position-indexed pairwise signal into a reusable lexical merging rule.

Under uniform random-order sampling at common length $n \geq 2$ and $b \geq 2$, the pair exposure $\rho_2(n, b) > 0$ is common to all position pairs. Multiplying the adjacent count table by this common factor scales all candidate scores equally and therefore does not change the merge ranking. We consequently use ordinary occurrence counts during vocabulary construction, yielding a merge rule that is independent of the denoising budget.

For $z^{(m)} = E_\tau(x^{(m)})$ and $n_m = |z^{(m)}|$, collect

$$c(u, v) = \sum_{m=1}^{M} \sum_{j=1}^{n_m - 1} \mathbf{1}\{z_j^{(m)} = u, z_{j+1}^{(m)} = v\}. \tag{10}$$

Let $c_+ = \sum_{u,v} c(u, v) > 0$, $c_{\mathrm{L}}(u) = \sum_v c(u, v)$, and $c_{\mathrm{R}}(v) = \sum_u c(u, v)$. The pooled distribution is $\widehat{\mu}(u, v) = c(u, v)/c_+$, with marginals $\widehat{\mu}_{\mathrm{L}} = c_{\mathrm{L}}/c_+$ and $\widehat{\mu}_{\mathrm{R}} = c_{\mathrm{R}}/c_+$.

From adjacent dependence to a merge score. To measure the cost of predicting adjacent token values independently, consider unrestricted joint and separate predictors shared across all occurrences. Their count-weighted negative log-likelihoods are

$$\mathcal{L}_{\mathrm{J}}(q) = -\sum_{u,v} c(u, v) \log q(u, v), \quad \mathcal{L}_{\mathrm{sep}}(q_{\mathrm{L}}, q_{\mathrm{R}}) = -\sum_{u,v} c(u, v) \log[q_{\mathrm{L}}(u) q_{\mathrm{R}}(v)].$$

Let $\mathcal{L}_{\mathrm{J}}^*$ and $\mathcal{L}_{\mathrm{sep}}^*$ denote their minima over normalized distributions. These are attained at $q^* = \widehat{\mu}$, $q_{\mathrm{L}}^* = \widehat{\mu}_{\mathrm{L}}$, and $q_{\mathrm{R}}^* = \widehat{\mu}_{\mathrm{R}}$, giving

$$\mathcal{L}_{\mathrm{sep}}^* - \mathcal{L}_{\mathrm{J}}^* = c_+ I_{\widehat{\mu}}(U; V) = \sum_{u,v} c(u, v) \log \frac{\widehat{\mu}(u, v)}{\widehat{\mu}_{\mathrm{L}}(u) \widehat{\mu}_{\mathrm{R}}(v)}. \tag{11}$$

We use each pair type's contribution to this gap as its merge score. Substituting the empirical probabilities yields

$$\widehat{\mathcal{W}}(u, v) = c(u, v) \log \frac{c_+ c(u, v)}{c_{\mathrm{L}}(u) c_{\mathrm{R}}(v)}. \tag{12}$$

Thus WPMI is occurrence count times pointwise mutual information, and its sum exactly equals the joint-versus-separate prediction gap on the adjacent-pair distribution. Restricting to adjacent pairs and aggregating their occurrences are construction choices that turn the pairwise analysis into a reusable lexical merge criterion. Appendix C.1 provides the derivation and distinguishes this objective from indexed dependence.

Vocabulary construction. Starting from a byte vocabulary, we repeatedly select the highest-scoring feasible pair, introduce a reversible token, and substitute left to right without overlaps. Counts and marginals are recomputed after each merge; retaining all entries gives $|V_{\tau_0}| + B$ entries after B additions (Appendix C.2).

What WPMI prioritizes. WPMI combines association with occurrence mass. A frequent pair scores zero when its joint probability equals the product of its marginals, while a pair's normalized contribution vanishes as its relative frequency goes to zero, even if its PMI diverges. The marginals come from the adjacent table rather than the single-token counts used by WordPiece. The score is independent of the denoising budget.

6 Experiments

We first examine compression and per-example encoded lengths before comparing held-out prediction loss and generation across denoising budgets. Corpus-level merge diagnostics evaluate the local signal used by the construction.

6.1 Experimental Setup

Training details. We conduct experiments on the One Billion Word Benchmark (LM1B) (Chelba et al., 2013) and OpenWebText (OWT) (Gokaslan et al., 2019). Following prior masked diffusion work (Sahoo et al., 2024), we use a 170M-parameter diffusion transformer with rotary positional embeddings. We train on LM1B for 1M steps with a sequence length of 128 tokens and on OWT for 100K steps with a sequence length of 1024 tokens. All runs use a batch size of 512. Evaluation uses the standard LM1B test split. For OWT, we reserve the last 100K documents as a held-out validation set following the same evaluation protocol. Appendix D.1 records the specified settings.

Evaluation metrics. We report bits per byte (BPB) by dividing the sequence-summed denoising loss by the total raw bytes and $\log 2$. BPB measures prediction loss conditional on encoded length. For zero-shot evaluation, we take models trained on OpenWebText and evaluate BPB on the validation splits of PTB (Marcus et al., 1993), WikiText (Merity et al., 2016), LM1B (Chelba et al., 2013), LAMBADA (Paperno et al., 2016), AG News (Zhang et al., 2015), and Scientific Papers (Xu et al., 2020), we detokenize each dataset and wrap sequences to 1024 tokens without interleaving end-of-sequence markers. Generative perplexity (Gen. PPL) is the exponential of the average reference-token negative log-likelihood of decoded samples, using GPT-2 Large (Radford et al., 2019) as the fixed evaluator. Lower values are better for both metrics. Compression ratio (CR) is the ratio of total raw bytes to total encoded tokens. Appendix D.2 specifies the evaluation and aggregation procedures.

Tokenizers. We compare byte-based BPE, WordPiece, Unigram and WPMI to analyze how tokenization affects language modeling and parallel generation. All tokenizers use 50,000 vocabulary entries on both LM1B and OWT. This fixes the vocabulary budget while allowing segmentation and compression to vary across methods.

Table 2: **Zero-shot BPB of OWT-trained models.** We evaluate each tokenizer on seven out-of-domain datasets without further training or adaptation. Lower is better, and the best result in each column is shown in bold.

Tokenizer	PTB	WikiText	LM1B	LAMBADA	AG News	PubMed	arXiv
BPE	**1.5249**	**1.1637**	**1.3829**	**1.4630**	1.4295	1.3212	1.4234
WordPiece	1.5768	1.2153	1.4356	1.5142	1.5420	1.3469	1.4951
Unigram	1.5566	1.1762	1.3910	1.4698	**1.4200**	1.3060	**1.4113**
WPMI	1.5488	1.1758	1.3984	1.4718	1.4590	**1.2833**	1.4162

6.2 Comparable compression and model fit

We first verify that the tokenizers provide a controlled comparison by measuring compression and held-out prediction quality. Comparable BPB provides a useful baseline, but does not eliminate differences in prediction error under the generation schedule.

BPE, WordPiece, and WPMI achieve similar compression on both datasets, with BPE-relative length gaps below 3%. Test BPB is also broadly comparable: on LM1B, all four methods fall within 1.1133–1.1603, and on OWT, BPE and WPMI are nearly identical at 1.0831 and 1.0850. WordPiece has a higher loss of 1.1307 on OWT despite similar compression, indicating that matching token budgets alone does not ensure matching model fit.

We further evaluate zero-shot BPB on seven datasets using the OWT-trained models (Table 2). BPE achieves the best overall zero-shot BPB, but the gaps among tokenizers remain modest. WordPiece, Unigram, and WPMI achieve broadly comparable results, suggesting similar model fit beyond the training corpus.

Together, the test and zero-shot results provide a model-fit baseline for the generation comparison. In particular, WPMI does not exhibit an overall advantage over BPE in either evaluation. We next examine whether broadly comparable model fit nevertheless leads to different generation quality as more tokens are sampled in parallel.

Table 1: **Compression and test BPB with 50K vocabulary entries.** CR is raw bytes per token; Gap is the BPE-relative length gap (%).

Tokenizer	CR	Gap (%)	Test BPB ↓
LM1B			
BPE	4.820	0.00	1.1133
WordPiece	4.810	0.56	1.1171
Unigram	4.638	4.70	1.1603
WPMI	4.735	2.35	1.1314
OWT			
BPE	4.493	0.00	1.0831
WordPiece	4.480	0.48	1.1307
Unigram	4.404	2.19	1.0783
WPMI	4.411	1.94	1.0850

6.3 Parallel Generation

We now examine whether their behavior diverges under parallel generation. Factorization error grows as the number of denoising steps K decreases, since more tokens must be predicted from the same context. Table 3 reports Gen. PPL and sequence entropy across denoising budgets K on OWT.

At aggressive parallelism ($K \leq 128$), WPMI achieves the lowest Gen. PPL, with the largest margin at $K = 32$ (118.83 vs. 138.83 for BPE). A crossover occurs around $K = 256$, beyond which BPE and WordPiece achieve lower Gen. PPL. This crossover partly reflects the slightly higher test BPB of WPMI (Table 1): at large K, factorization error diminishes and model fit becomes the dominant factor. Additionally, Gen. PPL is evaluated by GPT-2 Large, which itself uses a BPE tokenizer and may favor BPE-aligned subword boundaries, potentially amplifying BPE's advantage where generation quality differences are otherwise small.

Entropy supports a consistent picture. WPMI maintains the highest entropy from $K = 64$ onward, and since entropy does not depend on any specific tokenizer, this pattern suggests that the separation at low K reflects parallel generation quality rather than evaluator bias. The separation at small K and convergence at large K are consistent with the theoretical prediction in Section 4: factorization error is determined by tokenizer-induced interactions, so differences emerge only when parallel rounds are large enough to expose them.

Table 3: **Parallel generation across denoising budgets on OWT.** Gen. PPL and Entropy are reported for varying numbers of denoising steps K. Lower Gen. PPL and higher Entropy are better.

Tokenizer	Gen. PPL ↓							Entropy ↑						
K	16	32	64	128	256	512	1024	16	32	64	128	256	512	1024
BPE	235.62	138.83	98.28	78.20	64.08	**52.78**	42.88	**5.745**	**5.671**	5.604	5.541	5.460	5.384	5.284
WordPiece	236.73	144.60	102.65	78.72	**63.86**	53.16	**42.39**	5.737	5.667	5.602	5.520	5.446	5.376	5.273
Unigram	225.59	133.44	99.28	79.09	65.50	52.87	44.25	5.736	5.658	5.582	5.521	5.452	5.373	5.290
WPMI	**219.85**	**118.83**	**92.45**	**72.30**	67.16	55.28	45.32	5.742	5.670	**5.609**	**5.543**	**5.471**	**5.392**	**5.303**

Factorization error is directly shaped by tokenization, and this effect manifests as measurable differences in Gen. PPL and entropy even when compression and held-out prediction quality are comparable.

7 Related Work

Discrete diffusion and factorization analysis. Discrete diffusion models define categorical corruption processes and learn to reverse them through iterative denoising (Hoogeboom et al., 2021; Austin et al., 2021; Campbell et al., 2022). Masked diffusion models, which use an absorbing mask state, have shown particular promise for language modeling (Sahoo et al., 2024; Lou et al., 2023; Shi et al., 2024). A central challenge is factorization error from independent parallel sampling. Lavenant & Zanella (2025) decompose this error into a learned-conditional component and a schedule-dependent factorization cost, and Chen et al. (2025) study schedule-dependent error through information profiles. Both hold the token distribution fixed and vary the sampling policy. We instead fix the sampling policy and vary the tokenizer to study how the definition of tokens itself shapes factorization error.

Modeling and sampling dependencies. Several approaches aim to capture dependencies discarded by independent parallel sampling. Copula-based correction (Liu et al., 2025), residual-energy models (Xu et al., 2025), and latent-variable augmentation (Shariatian et al., 2025) introduce additional model components to approximate the joint distribution beyond factorized marginals. Orthogonally, dependency-aware samplers modify which positions are updated together: JYS (Park et al., 2025) optimizes the allocation of sampling timesteps, APD (Israel et al., 2026) reintroduces sequential conditioning via an autoregressive auxiliary model, and PUNT (Azangulov et al., 2025) selects approximately conditionally independent positions. Distillation methods offer yet another route to few-step generation (Hayakawa et al., 2024; Zhang et al., 2026; Yoo et al., 2026). All of these modify the model or the sampling procedure while operating on a fixed set of token variables.

Representation and tokenization. Tokenizer design has been studied primarily through compression and downstream performance (Bostrom & Durrett, 2020; Chizhov et al., 2024), and BLT (Pagnoni et al., 2025) groups bytes into variable-length patches using predictive entropy. More directly related to diffusion, Zonkey (Rozental, 2026) and Dynamic Chunking (Zhu et al., 2026) adapt segmentation or chunk structure for diffusion-based generation. Our work differs in providing a theoretical characterization of how tokenization affects parallel factorization error through an exact interaction-order decomposition.

8 Conclusion

We study tokenization as a source of factorization error in discrete diffusion language models. We show that lossless tokenizers can share the same oracle denoising objective while inducing different errors under parallel sampling, and derive an interaction-order decomposition that separates tokenizer-induced dependencies from schedule-dependent exposure. Empirically, tokenizers with similar compression and prediction quality exhibit substantially different behavior under aggressive parallel generation, with WPMI improving generation quality at small denoising budgets. These results suggest that tokenization should be treated as an integral part of discrete diffusion model design.

AI USE STATEMENT

We used generative AI tools to assist with writing, including refining grammar, improving sentence clarity, and polishing drafts written by the authors. These tools were also used to help review and debug experimental code. All AI-assisted outputs were reviewed by the authors, and we take full responsibility for the final content of this paper, including all text, claims, and experimental results.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of model architecture, training configurations including datasets, preprocessing, and hyperparameters, tokenizer construction procedures, and evaluation protocols in Appendix D. We will release the full codebase and pretrained checkpoints upon acceptance.

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.

Iskander Azangulov, Teodora Pandeva, Niranjani Prasad, Javier Zazo, and Sushrut Karmalkar. Parallel sampling from masked diffusion models via conditional independence testing. *arXiv preprint arXiv:2510.21961*, 2025.

Kaj Bostrom and Greg Durrett. Byte pair encoding is suboptimal for language model pretraining. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 4617–4624, 2020.

Andrew Campbell, Joe Benton, Valentin De Bortoli, Thomas Rainforth, George Deligiannidis, and Arnaud Doucet. A continuous time framework for discrete denoising models. *Advances in Neural Information Processing Systems*, 35:28266–28279, 2022.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. One billion word benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*, 2013.

Sitan Chen, Kevin Cong, and Jerry Li. Optimal inference schedules for masked diffusion models. *arXiv preprint arXiv:2511.04647*, 2025.

Pavel Chizhov, Catherine Arnett, Elizaveta Korotkova, and Ivan P Yamshchikov. Bpe gets picky: Efficient vocabulary refinement during tokenizer training. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 16587–16604, 2024.

Edo Dotan, Gal Jaschek, Tal Pupko, and Yonatan Belinkov. Effect of tokenization on transformers for biological sequences. *Bioinformatics*, 40(4):btae196, 2024.

Philip Gage. A new algorithm for data compression. *C Users Journal*, 12(2):23–38, 1994.

Aaron Gokaslan, Vanya Cohen, Ellie Pavlick, and Stefanie Tellex. Openwebtext corpus. `http://Skylion007.github.io/OpenWebTextCorpus`, 2019.

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

Timothy J Hazen and Fred Richardson. Modeling multiword phrases with constrained phrase trees for improved topic modeling of conversational speech. In *2012 IEEE Spoken Language Technology Workshop (SLT)*, pp. 222–227. IEEE, 2012.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. *Advances in neural information processing systems*, 34:12454–12465, 2021.

Daniel Israel, Guy Van den Broeck, and Aditya Grover. Accelerating diffusion llms via adaptive parallel decoding. *Advances in neural information processing systems*, 38:52870–52888, 2026.

Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 66–75, 2018.

Inception Labs, Samar Khanna, Siddhant Kharbanda, Shufan Li, Harshit Varma, Eric Wang, Sawyer Birnbaum, Ziyang Luo, Yanis Miraoui, Akash Palrecha, et al. Mercury: Ultra-fast language models based on diffusion. *arXiv preprint arXiv:2506.17298*, 2025.

Hugo Lavenant and Giacomo Zanella. Error bounds and optimal schedules for masked diffusions with factorized approximations. *arXiv preprint arXiv:2510.25544*, 2025.

Anji Liu, Oliver Broadrick, Mathias Niepert, and Guy Van den Broeck. Discrete copula diffusion. In *International Conference on Learning Representations*, volume 2025, pp. 88953–88979, 2025.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. *arXiv preprint arXiv:2310.16834*, 2023.

Mitch Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330, 1993.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

Artidoro Pagnoni, Ramakanth Pasunuru, Pedro Rodriguez, John Nguyen, Benjamin Muller, Margaret Li, Chunting Zhou, Lili Yu, Jason E Weston, Luke Zettlemoyer, et al. Byte latent transformer: Patches scale better than tokens. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9238–9258, 2025.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc-Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The lambada dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 1525–1534, 2016.

Yong-Hyun Park, Chieh-Hsin Lai, Satoshi Hayakawa, Yuhta Takida, and Yuki Mitsufuji. Jump your steps: Optimizing sampling schedule of discrete diffusion models. In *International Conference on Learning Representations*, volume 2025, pp. 96272–96300, 2025.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Alon Rozental. Zonkey: A hierarchical diffusion language model with differentiable tokenization and probabilistic attention. *arXiv preprint arXiv:2601.21768*, 2026.

Subham S Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 37:130136–130184, 2024.

Mike Schuster and Kaisuke Nakajima. Japanese and korean voice search. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 5149–5152. IEEE, 2012.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: long papers)*, pp. 1715–1725, 2016.

Dario Shariatian, Alain Durmus, Umut Simsekli, and Stefano Peluchetti. Latent-augmented discrete diffusion models. *arXiv preprint arXiv:2510.18114*, 2025.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. Simplified and generalized masked diffusion for discrete data. *Advances in neural information processing systems*, 37: 103131–103167, 2024.

Yuxuan Song, Zheng Zhang, Cheng Luo, Pengyang Gao, Fan Xia, Hao Luo, Zheng Li, Yuehang Yang, Hongli Yu, Xingwei Qu, et al. Seed diffusion: A large-scale diffusion language model with high-speed inference. *arXiv preprint arXiv:2508.02193*, 2025.

DiffusionGemma Team, Adrien Ali Taïga, James Assiene, Daniele Calandriello, Rahma Chaabouni, João Gante, Tamara von Glehn, Nate Keating, Chris Knutsen, Martin Kukla, et al. Diffusiongemma technical report. *arXiv preprint arXiv:2608.00146*, 2026.

Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu. Discourse-aware neural extractive text summarization. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pp. 5021–5031, 2020.

Minkai Xu, Tomas Geffner, Karsten Kreis, Weili Nie, Yilun Xu, Jure Leskovec, Stefano Ermon, and Arash Vahdat. Energy-based diffusion language models for text generation. In *International Conference on Learning Representations*, volume 2025, pp. 33769–33789, 2025.

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

Tunyu Zhang, Xinxi Zhang, Ligong Han, Haizhou Shi, Xiaoxiao He, Zhuowei Li, Hao Wang, Kai Xu, Akash Srivastava, Hao Wang, et al. T3d: Few-step diffusion language models via trajectory self-distillation with direct discriminative optimization. *arXiv e-prints*, pp. arXiv–2602, 2026.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

Kaiwen Zheng, Yongxin Chen, Hanzi Mao, Ming-Yu Liu, Jun Zhu, and Qinsheng Zhang. Masked diffusion models are secretly time-agnostic masked models and exploit inaccurate categorical sampling. In *International Conference on Learning Representations*, volume 2025, pp. 63186–63227, 2025.

Yichen Zhu, Xiaoming Shi, Peng Zhao, Weiyu Chen, Debing Zhang, and James Kwok. Dynamic chunking for diffusion language models. *arXiv preprint arXiv:2605.15676*, 2026.

A Oracle Loss, Sampling Error, and Decoding

All expectations use the raw-source law in Section 2. Length sums include only supported values. Length matching is not required in this section.

A.1 From Diffusion Time to Unmasking Schedules

For a supported masked input z_t with visible set O and $0 < \alpha_t < 1$, independent masking gives

$$q_t(z_t \mid z) = \alpha_t^{|O|}(1-\alpha_t)^{n-|O|}\mathbf{1}\{z_O = (z_t)_O\}.$$

The masking factor cancels in Bayes' rule. Thus, for $i \notin O$,

$$\mathbb{P}(Z_i = v \mid N_\tau = n, Z_t = z_t, t) = p_{\tau,n}(Z_i = v \mid Z_O = (z_t)_O). \tag{13}$$

The oracle therefore depends on the indexed visible tokens rather than diffusion time. At full masking it is the length-conditioned marginal.

Following Zheng et al. (2025), the continuous-time reverse process can be represented by successive single-position unmasking events. Each event reveals a uniformly selected masked position. Event times can be omitted for an oracle or another time-independent predictor; a time-dependent predictor retains them. The state is the full masked sequence, not its mask count alone. Grouping a uniform order into prescribed-size rounds gives the policies studied in Section 4. Independent predictions within a round use the same visible context. This batching is an approximation, not an exact time reparameterization.

A.2 Oracle Denoising Loss

Proof of Proposition 1. Fix a supported length $n \geq 1$. By equation 13, the unrestricted optimum at a masked target i is the conditional entropy $\mathrm{H}_{\tau,n}(Z_i \mid Z_O)$. Let $s = |O|$ and set $u = 1 - \alpha_t$. The integrated coefficient of this query in equation 2 is

$$\int_0^1 u^{n-s-1}(1-u)^s\,du = \frac{s!(n-s-1)!}{n!} = \frac{1}{n\binom{n-1}{s}}. \tag{14}$$

This is the probability that a uniform permutation places exactly O before i. Summing over targets and visible sets therefore averages the entropy chain rule and gives $\mathrm{H}_{\tau,n}(Z)$. The oracle integrand after the change of variables is bounded by $n \log |V_\tau|$, so finite source support also makes the length average finite.

Deterministic lossless encoding is injective on the source support, and sequence length is a function of the encoding. Hence

$$\mathrm{H}(X) = \mathrm{H}(Z^\tau) = \mathrm{H}(N_\tau) + \sum_n p_\tau^N(n)\,\mathrm{H}_{\tau,n}(Z).$$

Combining the identities proves

$$\mathcal{L}^*_{\mathrm{MDM}}(\tau) = \mathrm{H}(X) - \mathrm{H}(N_\tau).$$

The objective sums over positions. Per-token normalization defines a different quantity. □

Endpoints and length information. The identity uses the full-noise endpoints $\alpha_0 = 1$ and $\alpha_1 = 0$. If the oracle integral is restricted to $u \in [\varepsilon_0, 1-\varepsilon_1]$, where $\varepsilon_0, \varepsilon_1 \geq 0$ and $\varepsilon_0 + \varepsilon_1 < 1$, the same integrand bound gives

$$\begin{aligned} 0 &\leq \mathrm{H}(X) - \mathrm{H}(N_\tau) - \mathcal{L}^*_{\mathrm{MDM},\varepsilon}(\tau) \\ &\leq (\varepsilon_0 + \varepsilon_1)\mathbb{E}[N_\tau] \log |V_\tau|. \end{aligned} \tag{15}$$

This bounds omitted denoising loss, not a finite-endpoint ELBO with unchanged boundary terms. In particular, $\alpha_1 > 0$ does not give an exactly all-mask terminal law. Under per-example length matching, $\mathrm{H}(N_\tau) = \mathrm{H}(N)$ is common and $\mathrm{H}_{\tau,n}(Z) = \mathrm{H}(X \mid N = n)$. The length entropy vanishes only when the encoded length is constant almost surely.

A.3 Factorization Cost and Learned-Marginal Error

Proof of Proposition 2. Complete oracle conditionals at source-null histories by fixed normalized kernels. Successive summation over blocks in reverse order then normalizes the factorized law. Define

$$P_\tau(n, \mathcal{S}, z) = p_\tau^N(n)\Pi_n(\mathcal{S})p_{\tau,n}(z),$$
$$Q_{\tau,\Pi}(n, \mathcal{S}, z) = p_\tau^N(n)\Pi_n(\mathcal{S})\prod_k \prod_{i\in B_k} p_{\tau,n}(z_i \mid z_{O_k}). \tag{16}$$

The length and schedule factors cancel in the likelihood ratio. Using the block chain rule,

$$\begin{aligned}\mathcal{E}_\Pi(\tau) &= D_{\mathrm{KL}}(P_\tau \| Q_{\tau,\Pi}) \\ &= \mathbb{E}_{n,\mathcal{S}} \sum_k \left[\sum_{i\in B_k} \mathrm{H}_{\tau,n}(Z_i \mid Z_{O_k}) - \mathrm{H}_{\tau,n}(Z_{B_k} \mid Z_{O_k}) \right] \\ &= \mathbb{E}_{n,\mathcal{S}} \sum_k \mathrm{TC}_{\tau,n}(Z_{B_k} \mid Z_{O_k}).\end{aligned}$$

The joint block entropies sum to $\mathrm{H}_{\tau,n}(Z)$. Adding $\mathrm{H}(N_\tau)$ gives $\mathcal{J}_\Pi(\tau) = \mathrm{H}(X) + \mathcal{E}_\Pi(\tau)$. Singleton rounds have zero conditional total correlation and recover the exact probability chain rule, including the length-conditioned left-to-right factorization. □

For normalized learned conditionals q_i and a length model q_N, let

$$Q^q_{\tau,\Pi}(n, \mathcal{S}, z) = q_N(n)\Pi_n(\mathcal{S}) \prod_k \prod_{i\in B_k} q_i(z_i \mid n, \mathcal{S}, z_{O_k}).$$

Adding and subtracting the oracle log probabilities yields

$$\begin{aligned}D_{\mathrm{KL}}(P_\tau \| Q^q_{\tau,\Pi}) = \mathcal{E}_\Pi(\tau) &+ D_{\mathrm{KL}}(p_\tau^N \| q_N) \\ &+ \mathbb{E}_{n,\mathcal{S},Z} \sum_k \sum_{i\in B_k} D_{\mathrm{KL}}(p_{\tau,n}(Z_i \mid Z_{O_k}) \| q_i(\cdot \mid n, \mathcal{S}, Z_{O_k})).\end{aligned} \tag{17}$$

Here $(n, \mathcal{S}, Z) \sim P_\tau$. All additional terms are nonnegative and may be infinite under support mismatch. This identity concerns schedule-retaining process KL. It does not identify the training NELBO or reference-model generative perplexity with $\mathcal{E}_\Pi$.

B Tokenization-Dependent Factorization Error

B.1 The Common Conditional Source

The controlled comparison uses one encoded-length function on the common raw source:

$$N_\tau(x) = N(x) \qquad (\tau \in \mathfrak{T},\ x \in \mathrm{supp}(p_X)). \tag{18}$$

Every encoder therefore conditions on the same raw population at $N = n$, although its pushforward $p_{\tau,n}$ may differ. Equal length distributions alone are insufficient because they may assign lengths to different raw examples. At a common length, the position canvas, prescribed round sizes, and sampling weights are shared. For Π_b, the number of rounds is $\lceil n/b \rceil$. Neither identical coverage fluctuations nor equal wall-clock cost is implied.

B.2 Proof of the Interaction Decomposition

Fix $N = n \geq 2$ and suppress conditioning. The proof allows any prescribed positive round sizes under a uniform random order. Write $s_k = |O_k|$ and $m_k = |B_k|$. For nonempty subsets, the signed co-information convention and its inversion are

$$I_{|A|}(Z_A) = \sum_{\varnothing \neq D \subseteq A} (-1)^{|D|+1} \mathrm{H}(Z_D), \tag{19}$$

$$\mathrm{H}(Z_D) = \sum_{\varnothing \neq A \subseteq D} (-1)^{|A|+1} I_{|A|}(Z_A). \tag{20}$$

Thus $I_1(Z_i) = \mathrm{H}(Z_i)$, $I_2(Z_i, Z_j) = I(Z_i; Z_j)$, and $I_3(Z_i, Z_j, Z_\ell) = I(Z_i; Z_j) - I(Z_i; Z_j \mid Z_\ell)$. The inversion follows by substitution: the coefficient of $\mathrm{H}(Z_C)$ is

$$\sum_{C \subseteq A \subseteq D} (-1)^{|A|+|C|} = \mathbf{1}\{C = D\}.$$

Proof of Theorem 3. For a fixed schedule, entropy inversion gives

$$\mathrm{H}(Z_i \mid Z_{O_k}) = \sum_{\substack{A \subseteq O_k \cup \{i\} \\ i \in A}} (-1)^{|A|+1} I_{|A|}(Z_A).$$

Let $k_A = \max\{k : A \cap B_k \neq \varnothing\}$. The interaction on A appears exactly once in the prediction entropy sum when $|A \cap B_{k_A}| = 1$: its last position is predicted after all other positions in A. It does not appear when the last occupied round contains at least two positions in A. Subtracting the joint entropy therefore yields

$$\begin{aligned} &D_{\mathrm{KL}}\Big(p_{\tau,n}\Big\|p^{\|}_{\tau,\mathcal{S}}(\cdot \mid n)\Big) \\ &\quad= \sum_k \sum_{i \in B_k} \mathrm{H}(Z_i \mid Z_{O_k}) - \mathrm{H}(Z) \\ &\quad= \sum_{\substack{A \subseteq [n] \\ |A| \geq 2}} (-1)^{|A|} \mathbf{1}\{|A \cap B_{k_A}| \geq 2\} I_{|A|}(Z_A). \end{aligned} \tag{21}$$

These are signed coefficients, not separate nonnegative penalties.

For a fixed r-subset A, its permutation ranks are a uniform r-subset of $[n]$. Exactly $m_k\binom{s_k}{r-1}$ rank sets have one member in round k and all other members earlier. Hence, for the fixed-b size vector,

$$\rho_r(n, b) = \mathbb{P}_{\mathcal{S}}(|A \cap B_{k_A}| \geq 2) = 1 - \frac{\sum_k m_k \binom{s_k}{r-1}}{\binom{n}{r}}. \tag{22}$$

The probability depends on r and the size vector, not on A. Averaging equation 21 gives

$$\mathcal{E}_{\Pi_b}(\tau \mid n) = \sum_{r=2}^{n} (-1)^r \rho_r(n, b) S_r(\tau).$$

No interaction orders are truncated. □

For $r = 2$, the event is co-generation of the two positions. With $\ell = n - b\lfloor n/b \rfloor$,

$$\rho_2(n, b) = \frac{\sum_k \binom{m_k}{2}}{\binom{n}{2}} = \frac{b-1}{n-1} - \frac{\ell(b-\ell)}{n(n-1)}. \tag{23}$$

This includes a shortened last round and remains valid for $b \geq n$. Singleton sampling has $\rho_r(n, 1) = 0$. A single all-token round has $\rho_r = 1$ for $r \geq 2$ and error $\mathrm{TC}(Z)$.

Weaker control from equal compression. On a common source with $\mathbb{E}|X| > 0$, define $\mathrm{CR}(\tau) = \mathbb{E}|X|/\mathbb{E}N_\tau$. Equal compression fixes only the mean encoded length $\mathbb{E}N_\tau = \nu > 0$. For integer $b \geq 1$ and $K_\tau = \lceil N_\tau / b \rceil$,

$$\frac{\nu}{b} \leq \mathbb{E}K_\tau \leq \frac{\nu + b - 1}{b}, \qquad |\mathbb{E}K_\tau - \mathbb{E}K_{\tau'}| \leq \frac{b-1}{b} < 1. \tag{24}$$

Indeed, $U_\tau = bK_\tau - N_\tau$ satisfies $0 \leq U_\tau \leq b-1$ and $b\mathbb{E}K_\tau = \nu + \mathbb{E}U_\tau$. For unequal mean lengths, the same argument gives

$$|\mathbb{E}K_\tau - \mathbb{E}K_{\tau'}| \leq \frac{|\mathbb{E}N_\tau - \mathbb{E}N_{\tau'}| + b - 1}{b}. \tag{25}$$

These are average-call bounds for fixed b.

B.3 Pairwise Calibration and Tokenizer Comparisons

Fix $N = n \geq 2$ and define

$$h_\tau(s) = \mathbb{E}_{i,O:\,|O|=s}\,\mathrm{H}(Z_i^\tau \mid Z_O^\tau), \qquad d_\tau(s) = h_\tau(s) - h_\tau(s+1). \tag{26}$$

The target is uniform and the visible set excludes it. The ranges are $0 \leq s \leq n-1$ for h_τ and $0 \leq s \leq n-2$ for d_τ. Adding a uniformly selected hidden position gives

$$d_\tau(s) = \mathbb{E}_{i,j,O:\,|O|=s} I(Z_i^\tau; Z_j^\tau \mid Z_O^\tau) \geq 0, \qquad d_\tau(0) = \frac{S_2(\tau)}{\binom{n}{2}}.$$

Thus $g_\tau(s) = d_\tau(s)/d_\tau(0)$ when $S_2(\tau) > 0$.

For prescribed round sizes, set $s_k = |O_k|$, $s_{K+1} = n$, and $a_\Pi(s) = s_{k+1} - 1 - s$ for $s_k \leq s < s_{k+1}$. All sums over s below range from 0 to $n-2$. The entropy chain rule averaged over a uniform order gives $\sum_{s=0}^{n-1} h_\tau(s) = \mathrm{H}(Z^\tau)$ and

$$\mathcal{E}_\Pi(\tau \mid n) = \sum_k |B_k| h_\tau(s_k) - \mathrm{H}(Z^\tau) = \sum_s a_\Pi(s) d_\tau(s). \tag{27}$$

This is the random-order information-profile identity of Lavenant & Zanella (2025). Within each round,

$$|B_k| h(s_k) - \sum_{j=s_k}^{s_{k+1}-1} h(j) = \sum_{s=s_k}^{s_{k+1}-2} (s_{k+1} - 1 - s) d(s).$$

For fixed b, write $a_b = a_{\Pi_b}$ and $\mathcal{P}_b(\tau \mid n) := \rho_2(n,b) S_2(\tau)$. Since $\sum_s a_b(s) = \sum_k \binom{|B_k|}{2} = \rho_2(n,b)\binom{n}{2}$, for $b \geq 2$ and $S_2(\tau) > 0$,

$$\frac{\mathcal{E}_{\Pi_b}(\tau \mid n)}{\mathcal{P}_b(\tau \mid n)} = \frac{\sum_s a_b(s) g_\tau(s)}{\sum_s a_b(s)}. \tag{28}$$

Proof of Proposition 4. For positive $S_2(\tau_0), S_2(\tau_1)$, define

$$\epsilon_b(\tau_1, \tau_0) = \frac{\sum_s a_b(s)[g_{\tau_1}(s) - g_{\tau_0}(s)]_+}{\sum_s a_b(s) g_{\tau_0}(s)}, \qquad [x]_+ = \max\{x, 0\}. \tag{29}$$

The denominator is positive because $g_{\tau_0}(0) = 1$ and $a_b(0) = \min\{b, n\} - 1 > 0$. Equation equation 27 gives

$$\begin{aligned} \frac{\mathcal{E}_{\Pi_b}(\tau_1 \mid n)}{\mathcal{E}_{\Pi_b}(\tau_0 \mid n)} &= \frac{S_2(\tau_1)}{S_2(\tau_0)} \frac{\sum_s a_b(s) g_{\tau_1}(s)}{\sum_s a_b(s) g_{\tau_0}(s)} \\ &\leq \frac{S_2(\tau_1)}{S_2(\tau_0)} [1 + \epsilon_b(\tau_1, \tau_0)]. \end{aligned}$$

The inequality follows from $g_1(s) \leq g_0(s) + [g_1(s) - g_0(s)]_+$ and $a_b(s) \geq 0$. If $S_2(\tau_0)/S_2(\tau_1) - 1 > \epsilon_b(\tau_1, \tau_0)$, the upper bound is strictly below one. □

Identical normalized responses give proportional errors even when equation 28 is far from one. When $S_2 = 0$, use the unnormalized identity equation 27. With variable common lengths, apply the comparison within each stratum and average error inequalities, not unweighted ratios.

Relation to interaction orders and schedules. Expanding conditional mutual information and averaging yields

$$d_\tau(s) = \sum_{r=2}^{s+2} (-1)^r \binom{s}{r-2} \frac{S_r(\tau)}{\binom{n}{r}}. \tag{30}$$

Indeed, $I(Z_i; Z_j \mid Z_O) = \sum_{D \subseteq O} (-1)^{|D|} I_{|D|+2}(Z_i, Z_j, Z_D)$. The transform is triangular. For positive S_2, identical normalized responses are therefore equivalent to identical S_r/S_2 for every $r \geq 2$. Moreover, $d_{\tau_1}(s) \leq d_{\tau_0}(s)$ for every s is equivalent to error dominance for every prescribed-size random-order policy. Sufficiency follows from equation 27; necessity follows by using s singleton rounds, one two-token round, and singleton rounds thereafter. This equivalence ranges over all prescribed size patterns, not only the fixed-b family.

Pair-specific, shared, and synergistic information. If $S_r = 0$ for all $r \geq 3$, then equation 30 gives $d(s) = d(0)$ and $\mathcal{E}_{\Pi_b} = \mathcal{P}_b$. In contrast, let $Z_i = (T, X_i)$ with mutually independent components. Then $S_r = \binom{n}{r} \mathrm{H}(T)$ for $r \geq 2$, $d(0) = \mathrm{H}(T)$, and $d(s) = 0$ for $s \geq 1$. Only the first round incurs a cost for the shared variable, so

$$\begin{aligned} \mathcal{E}_{\Pi_b}(\tau \mid n) &= (\min\{b, n\} - 1)\,\mathrm{H}(T), \\ \mathcal{P}_b(\tau \mid n) &= \sum_k \binom{|B_k|}{2} \mathrm{H}(T). \end{aligned} \tag{31}$$

When $b \mid n$, $2 \leq b \leq n$, and $\mathrm{H}(T) > 0$, the ratio $\mathcal{P}_b / \mathcal{E}_{\Pi_b}$ is $n/2$. Thus pairwise dependence can substantially overcount shared information without invalidating comparisons between representations with the same normalized response. Conversely, four fair bits constrained to even parity have independent proper subsets and $S_2 = 0$, but two rounds of size two give $\mathcal{E}_{\Pi_2} = \log 2$. This case is outside the positive-S_2 normalization of the comparison proposition.

C WPMI Construction and Its Limits

C.1 Shared Prediction on Adjacent Counts

For the ordinary adjacent table in equation 10, let $\widehat{\mu}(u, v) = c(u, v)/c_+$ with $c_+ > 0$. Consider unrestricted shared joint and separate predictors:

$$\begin{aligned} \mathcal{L}_{\mathrm{J}}(q) &= -\sum_{u,v} c(u, v) \log q(u, v), \\ \mathcal{L}_{\mathrm{sep}}(q_{\mathrm{L}}, q_{\mathrm{R}}) &= -\sum_{u,v} c(u, v) \log[q_{\mathrm{L}}(u) q_{\mathrm{R}}(v)]. \end{aligned}$$

Their empirical fits are $\widehat{\mu}$, $\widehat{\mu}_U$, and $\widehat{\mu}_V$. Consequently,

$$\mathcal{L}^*_{\mathrm{sep}} - \mathcal{L}^*_{\mathrm{J}} = c_+ I_{\widehat{\mu}}(U; V) = \sum_{u,v} \widehat{\mathcal{W}}(u, v). \tag{32}$$

WPMI is the contribution of one cell to this contrast. Individual contributions may be negative although their sum is nonnegative. The factor c_+ inside the logarithm in equation 12 cannot be dropped: its contribution $c(u, v) \log c_+$ varies across candidate pairs.

Indexed and pooled dependence. Let $\Lambda = \mathbb{E}[(N_\tau - 1)_+] > 0$ and define the population occurrence law by

$$\mathbb{E}_\mu f(J, U, V) = \frac{1}{\Lambda} \mathbb{E}_X \sum_{j=1}^{N_\tau - 1} f((N_\tau, j), Z_j, Z_{j+1}). \tag{33}$$

The index $J = (N_\tau, j)$ retains length and position. Then

$$\Lambda I_\mu(U; V \mid J) = \sum_n p^N_\tau(n) \sum_{j=1}^{n-1} I_{\tau,n}(Z_j; Z_{j+1}),$$

whereas a shared predictor without access to J gives the pooled contrast $\Lambda I_\mu(U; V)$. The difference is

$$I_\mu(U; V \mid J) - I_\mu(U; V) = I_\mu((U, V); J) - I_\mu(U; J) - I_\mu(V; J), \tag{34}$$

which may have either sign. Uniform sampling of positions does not equate indexed and pooled mutual information. The population cell contribution is

$$\mathcal{W}(u, v) = \Lambda \mu(u, v) \log \frac{\mu(u, v)}{\mu_U(u) \mu_V(v)}. \tag{35}$$

For the empirical law of M equally weighted examples, $\Lambda = c_+/M$ and $\widehat{\mathcal{W}} = M\mathcal{W}$.

Fixed-canvas interpretation. Multiplying an entire count table by a constant $\rho > 0$ preserves its fitted probabilities and multiplies every WPMI score by ρ. At common length $n \geq 2$ and $b \geq 2$, the pair exposure $\rho_2(n, b)$ is such a common factor and does not affect ranking. With variable document lengths, it is not common across occurrences. The construction uses ordinary document-adjacent counts. Fixed LM context lengths do not make construction documents length-matched, and packing may change which boundaries are included.

C.2 Construction at a Fixed Vocabulary Budget

Algorithm 1 Ordinary-count WPMI construction

Require: Raw corpus, lossless byte encoder, B additions, fixed eligibility and tie rules
Encode the corpus and initialize an empty ordered merge list
for $t = 1, \ldots, B$ **do**
    Form the full ordinary adjacent table and its row and column sums
    Form the feasible positive-count pairs $\mathcal{A}(\tau)$
    if $\mathcal{A}(\tau) = \varnothing$ **then**
        return Current encoder marked as incomplete
    end if
    Select the maximal score in equation 12 using deterministic ties
    Add a fresh reversible token and substitute left to right without overlaps
    Record the rule and update segmentation and counts
end for
return Encoder and ordered merge list

All existing vocabulary entries are retained, so B successful additions give $|V_{\tau_0}| + B$ entries. Encoding replays the ordered rules, and decoding expands token spellings. Special-token accounting follows the same vocabulary-budget convention across methods.

The full adjacent table is formed before candidate filtering. Eligibility specifies permitted boundaries, maximum spelling length, and duplicate-spelling rules. A run of q identical tokens has $q - 1$ self-pair edges but only $\lfloor q/2 \rfloor$ nonoverlapping substitutions. Score counts therefore need not equal replacement counts. Negative scores alone do not justify stopping below a fixed vocabulary budget. A greedy dead end is reported rather than filled with dummy entries.

D Experimental Details

D.1 Training details

Data and preprocessing. For LM1B, we use 30,301,028 training and 306,688 test sentences, applying the MDLM detokenization procedure before tokenizer fitting and model training. For OWT, we reserve the final 100K documents for validation and use the preceding 7,913,769 documents for training. Tokenizers are fitted exclusively on training data. All methods use a 50K vocabulary including five shared auxiliary tokens (`[BOS]`, `[EOS]`, `[PAD]`, `[MASK]`, and `[UNK]`). The text frontend uses the GPT-2 ByteLevel pretokenizer and its reversible 256-byte alphabet, without Unicode normalization or lowercasing; vocabulary entries do not cross pretokenization boundaries.

Tokenizer construction and encoding. We compare BPE, WordPiece, Unigram, and WPMI. BPE merges the most frequent adjacent pair. Unigram learns piece probabilities and prunes its vocabulary according to a unigram likelihood objective. WordPiece selects the pair maximizing the normalized association $c(u, v)/(c(u)c(v))$ (Dotan et al., 2024). WPMI uses the count-weighted adjacent-PMI criterion derived in Section 5. Merge-based methods exclude pairs with fewer than two occurrences, update counts after each merge, apply nonoverlapping substitutions, and resolve ties deterministically. At inference, BPE applies its ordered merge rules, Unigram uses maximum-probability segmentation, and WordPiece and WPMI share greedy longest-match encoding.

Model and optimization. We use a 170M-parameter noncausal diffusion transformer with 12 blocks, hidden width 768, 12 attention heads, FFN width 3072, rotary position embeddings, GELU

activations, dropout 0.1, and untied input and output embeddings. All models use the continuous-time absorbing-mask SUBS parameterization without time conditioning. LM1B models are trained at sequence length 128 for 1M updates, while OWT models use length 1024 for 100K updates. Optimization uses AdamW with learning rate 3×10^{-4}, $(\beta_1, \beta_2) = (0.9, 0.999)$, zero weight decay, 2,500 linear warmup steps, and global gradient clipping at 1.0. We use a global batch size of 512, bfloat16 mixed precision, EMA with decay 0.9999, and training seed 1.

D.2 METRICS

For evaluation examples $\{x^{(m)}\}_{m=1}^{M}$, let $|x^{(m)}|$ denote the raw-byte count.

Bits per byte (BPB). BPB measures the estimated sequence-summed denoising loss $\widehat{\ell}_{\mathrm{MDM}}$ in bits per raw byte:

$$\mathrm{BPB} = \frac{\sum_m \widehat{\ell}_{\mathrm{MDM}}(x^{(m)}; \theta, \tau)}{(\log 2) \sum_m |x^{(m)}|}. \tag{36}$$

Generative perplexity. Let $y^{(s)}$ denote generated samples tokenized for a fixed reference language model r. Generative perplexity is

$$\mathrm{GenPPL} = \exp\left(-\frac{\sum_s \sum_{j=1}^{|y^{(s)}|} \log r(y_j^{(s)} \mid y_{<j}^{(s)})}{\sum_s |y^{(s)}|} \right). \tag{37}$$

Compression and length gap. Compression measures raw bytes per token. The length gap measures the total absolute token-count difference from BPE as a percentage of BPE's total count. Token counts exclude padding and auxiliary markers.

$$\mathrm{CR}(\tau) = \frac{\sum_m |x^{(m)}|}{\sum_m N_\tau(x^{(m)})}, \mathrm{Gap}(\tau) = 100 \frac{\sum_m |N_\tau(x^{(m)}) - N_{\mathrm{BPE}}(x^{(m)})|}{\sum_m N_{\mathrm{BPE}}(x^{(m)})}. \tag{38}$$