DUEL: EXACT LIKELIHOOD FOR MASKED DIFFUSION REVEALS IMPROVED PRE-TRAINING SCALING LAWS

Anonymous authors
Paper under double-blind review

ABSTRACT

Masked diffusion models (MDMs) are trained to predict tokens in every order but generate in one. As such, MDM likelihood is an intractable sum over all orders. The evidence lower bound (ELBO) is used as a proxy, but it is a loose bound and assumes positions are unmasked uniformly at random. Frontier MDMs, however, select positions with deterministic rules such as confidence. To resolve this mismatch, we introduce DUEL, a framework that unifies common deterministic MDM samplers. **DUEL allows for likelihood computation that is both exact and under the sampler that is actually used**: replay generation, reveal the true tokens instead of sampled ones, and sum their log-probabilities. DUEL is a simple plug-and-play algorithm applicable to any pretrained MDM with no retraining. Evaluating with DUEL instead of the ELBO **changes what we conclude about MDMs**. The perplexity gap between MDMs and autoregressive (AR) models shrinks by up to 32% in-domain and 82% zero-shot. In scaling laws, the reported $16\times$ MDM–AR compute gap is $9\times$ under DUEL: between $1/3$ and $1/2$ of it was an evaluation artifact. Scoring multiple-choice answers with DUEL improves LLaDA-8B accuracy on all six tasks, by up to 9.7 points. DUEL also selects samplers correctly: it matches a gold-standard LLM judge on *every* decided comparison, which generative perplexity and MAUVE fail to do. DUEL reveals that **MDMs are ultimately more competitive** with AR models than the ELBO suggests.

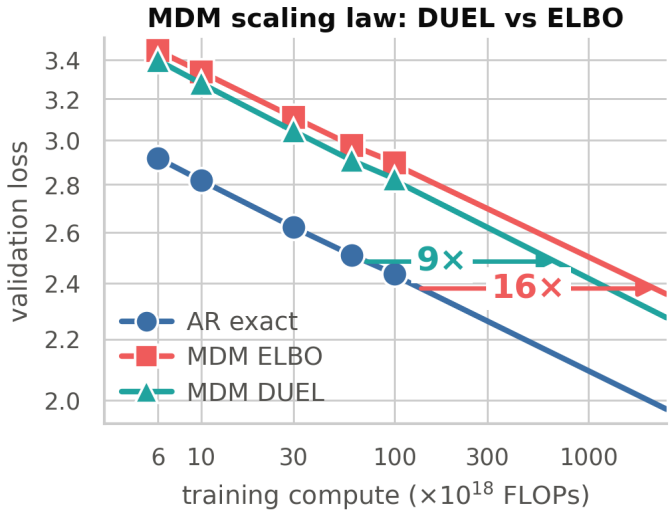

Figure 1: **DUEL shrinks the MDM–AR compute gap from $16\times$ to $9\times$.** Both MDM curves measure the same checkpoints, scored by the ELBO *bound* or the *exact* DUEL likelihood. Compute optimal frontiers fitted over 6–100 $\times$ 10^{18} FLOPs and extrapolated; arrows show how much more compute an MDM needs to reach AR's loss. Full numbers in Table 2.

1 INTRODUCTION

Masked diffusion models (MDMs) bring diffusion to text (Austin et al., 2021; Sahoo et al., 2024; Shi et al., 2024; Lou et al., 2024) and now scale to billions of parameters with strong benchmark results (Nie et al., 2025; Bie et al., 2025; Inception Labs et al., 2025; Song et al., 2025). Unlike autoregressive models, an MDM can generate several tokens per forward pass, in any order, and can be steered by guidance at inference time (Schiff et al., 2024).

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.

Its likelihood, however, is harder to compute. Following Kim et al. (2025a), an autoregressive model is *order-aware*: its generation order is fixed as left-to-right, so its likelihood factorizes along it and is exact. An MDM is *order-agnostic* (Uria et al., 2014; Kim et al., 2025a): it is trained to reveal tokens in every order but generates in only one. Its likelihood therefore marginalizes over all unmasking orders, an intractable sum with more than $L!$ terms.

In practice, the evidence lower bound (ELBO) is used as a tractable proxy for MDM likelihood (also its training objective). This proxy has two problems. First, it is only a lower bound: by Jensen's inequality, the expected log-likelihood over unmasking orders lies below the log of the expected likelihood, and this gap is large when orders differ widely in how well they explain a sequence. Second, it assumes that tokens are unmasked uniformly at random during inference time. However, frontier MDMs such as LLaDA, Dream, LLaDA 2.0, SDAR and Nemotron-Diffusion choose positions by a deterministic rule, such as revealing the most confident positions first (Nie et al., 2025; Ye et al., 2025; Bie et al., 2025; Cheng et al., 2025; Fu et al., 2026). The ELBO is thus a bound on the likelihood for a sampling procedure that is rarely run.

We introduce the Deterministic Unmasking Exact Likelihood (DUEL) framework to resolve this. A DUEL sampler pairs a pretrained denoiser with a deterministic unmasking rule to generate text. The rule fixes only which positions to reveal; the tokens at those positions are still sampled from the denoiser. Likelihood normally marginalizes over all unmasking orders, a sum with more than $L!$ terms. A deterministic rule collapses this sum to a single term that mirrors the generation procedure: replay generation on the sequence, revealing the true tokens instead of sampled ones, and multiply the denoiser's probabilities along the way. DUEL is a plug-and-play evaluation methodology: it computes proper likelihood for a standard MDM under its actual test-time policy, with no retraining or architectural changes, and it corrects how MDMs are measured, not the models themselves.

Accurate likelihood is crucial to evaluating MDMs: it shapes our understanding of their capabilities, shortcomings and cost, especially in comparison to autoregressive models. Evaluating MDMs with DUEL instead of the ELBO changes that understanding, revealing MDMs to be more competitive than previously thought along several crucial axes. Our contributions are as follows:

- **DUEL.** We formalize DUEL samplers, unifying popular deterministic MDM samplers, and prove their likelihood is exact and computable (Theorem 1).
- **Perplexity gap.** The MDM–AR perplexity gap shrinks by 12–32% in-domain and 22–82% zero-shot across eight datasets and four models (SEDD, MDLM, BD3-LM, LLaDA-8B).
- **Scaling laws.** Along the compute-optimal frontier, the compute an MDM needs to match AR loss falls from roughly $16\times$ under the ELBO to $9\times$ under DUEL (Nie et al., 2024), a 31–41% cut at every budget.
- **Multiple-choice benchmarks.** Scoring answers with DUEL, not the ELBO, lifts LLaDA-8B accuracy on all six tasks, by up to 9.7 points (2.5 and 4.2 on average under two samplers).
- **Sampler selection.** The sampler sets text quality; DUEL agrees with a gold-standard LLM judge on $14/14$ decided comparisons, against $8/14$ for generative perplexity and $12/14$ for MAUVE.

2 Preliminaries

2.1 Masked Diffusion Models

Data and masks. A sequence is $\boldsymbol{x} = (x^1, \ldots, x^L) \in \mathcal{V}^L$ over a vocabulary $\mathcal{V}$. During generation only some positions are known; for a set $S \subseteq [L]$ of positions we write $\boldsymbol{x}^S \triangleq (x^\ell)_{\ell \in S}$ for the tokens at S and call it a *partially revealed* sequence. It records S, hence also the masked positions $[L] \setminus S$.

Forward process. The forward process q masks each position independently with probability $1 - \alpha_t$, where $\alpha_0 = 1$ and $\alpha_1 = 0$ (nothing to all masked): $q_t(\boldsymbol{z}_t^\ell \mid \boldsymbol{x}) = \mathrm{Cat}\left(\alpha_t\, \boldsymbol{e}_{x^\ell} + (1-\alpha_t)\, \boldsymbol{e}_{\mathrm{m}}\right)$. Masking is absorbing: $\boldsymbol{z}_t$ is a partially revealed sequence with revealed set $S(\boldsymbol{z}_t)$ shrinking in t.

Reverse process. Given the clean sequence, the forward process reverses exactly: going from time t to $s < t$, the posterior $q_{s|t}(\boldsymbol{z}_s \mid \boldsymbol{z}_t, \boldsymbol{x})$ keeps every revealed position and reveals each masked position with probability $(\alpha_s - \alpha_t)/(1 - \alpha_t)$, at its clean value (Appendix B). At generation time

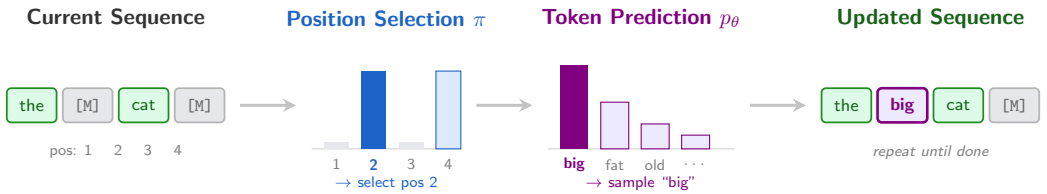

Figure 2: **One step of MDM generation.** Given a partially masked sequence, the unmasking policy π performs *position selection* (choosing position 2), and the denoiser p_θ performs *token prediction* at that position (sampling "big"). This repeats until no masked positions remain.

$\boldsymbol{x}$ is unknown. A network, the *denoiser*, takes a partially revealed sequence and outputs, for every masked position $\ell \notin S$, a categorical distribution $p_\theta(x^\ell \mid \boldsymbol{x}^S)$ over the vocabulary predicting the clean token there. The generative model substitutes these predictions for the unknown clean sequence, $p_\theta(\boldsymbol{z}_s \mid \boldsymbol{z}_t) = q_{s|t}(\boldsymbol{z}_s \mid \boldsymbol{z}_t, \boldsymbol{x} \leftarrow p_\theta)$, and runs from the fully masked sequence at $t = 1$ to $t = 0$.

Training objective. The likelihood $p_\theta(\boldsymbol{x})$ marginalizes over the latent trajectory and is intractable. MDMs are trained on its evidence lower bound (ELBO), a weighted cross-entropy at masked positions (Sahoo et al., 2024; Shi et al., 2024; Ou et al., 2024):

$$-\log p_\theta(\boldsymbol{x}) \;\le\; \mathbb{E}_{t,\,\boldsymbol{z}_t}\Big[\, w_t \sum_{\ell \notin S(\boldsymbol{z}_t)} -\log p_\theta\big(x^\ell \mid \boldsymbol{x}^{S(\boldsymbol{z}_t)}\big)\Big], \qquad w_t = \frac{-\alpha_t'}{1-\alpha_t}, \tag{1}$$

with $t \sim \mathrm{Unif}[0,1]$ and $\boldsymbol{z}_t \sim q_t(\cdot \mid \boldsymbol{x})$: one denoiser is trained on every masking level at once.

2.2 Masked Diffusion as Any-Order Autoregression

The diffusion formulation of Section 2.1 hides a simpler description: generation reveals the positions of $\boldsymbol{x}$ a few at a time, in some order. We make the order explicit: an MDM is an any-order autoregressive model (AO-ARM), a class originating with order-agnostic NADE (Uria et al., 2014) and shown equivalent by Hoogeboom et al. (2021a); Ou et al. (2024); Kim et al. (2025a).

Definition 1 (Ordered partition)**.** An *ordered partition* of $[L] = \{1, \ldots, L\}$ is a tuple $\sigma = (\sigma_1, \ldots, \sigma_T)$ of non-empty, disjoint subsets with $\bigcup_{t=1}^{T} \sigma_t = [L]$ and $\sigma_t \cap \sigma_{t'} = \emptyset$.

Ordered partitions. An ordered partition describes the order in which an MDM unmasks tokens: σ_t is the set of positions revealed at step t. We write $\sigma_{<t} = \sigma_1 \cup \cdots \cup \sigma_{t-1}$ for the positions revealed before step t, so that the sampler's state before that step is the partially revealed sequence $\boldsymbol{x}^{\sigma_{<t}}$. Ordered partitions capture both sequential and parallel unmasking, a distinguishing feature of MDMs over autoregressive models: when every part is a singleton we recover sequential unmasking with $T = L$ steps and σ is a permutation; otherwise we have parallel unmasking with $T < L$ steps, possible because the denoiser predicts every masked position in one forward pass.

Joint factorization. With the order made explicit, an MDM sampler alternates two operations visualized in Figure 2. It *selects* which masked positions to reveal next, and it *predicts* the tokens at those positions. Position selection is governed by an *unmasking policy* $\pi(\sigma_t \mid \boldsymbol{x}^{\sigma_{<t}})$, a distribution over non-empty subsets of the still-masked positions. The policy may depend on the tokens revealed so far, $\boldsymbol{x}^{\sigma_{<t}}$, and therefore also on the denoiser's predictions computed from them. Token prediction is done by the denoiser: for each selected position $\ell \in \sigma_t$ it draws a token from $p_\theta(\cdot \mid \boldsymbol{x}^{\sigma_{<t}})$. The joint probability of generating $\boldsymbol{x}$ via σ factorizes into these two operations at each step (Proposition 1):

$$p_\theta(\boldsymbol{x}, \sigma) = \prod_{t=1}^{T} \underbrace{\pi\big(\sigma_t \mid \boldsymbol{x}^{\sigma_{<t}}\big)}_{\text{position selection}} \cdot \underbrace{\prod_{\ell \in \sigma_t} p_\theta\big(x^\ell \mid \boldsymbol{x}^{\sigma_{<t}}\big)}_{\text{token prediction}}. \tag{2}$$

The inner product over $\ell \in \sigma_t$ is the parallel prediction of the tokens selected at step t: one forward pass yields a distribution for every masked position, and the sampler draws the tokens at σ_t

Algorithm 1 DUEL sampling

Require: denoiser p_θ, rule F
$S \leftarrow \emptyset$ ▷ *nothing revealed*
while masked tokens remain **do**
  $\boldsymbol{P} \leftarrow p_\theta(\cdot \mid \boldsymbol{x}^S)$ ▷ *one forward pass*
  $I \leftarrow F(\boldsymbol{x}^S)$ ▷ *positions to reveal*
  for $\ell \in I$ **do**
    $x^\ell \sim \mathrm{Cat}(P_\ell)$ ▷ *sample a token*
  $S \leftarrow S \cup I$
return $\boldsymbol{x}$ ▷ *a sample from* $p_\theta^{\pi^F}$

Algorithm 2 DUEL exact log-likelihood

Require: sequence $\boldsymbol{x}$, denoiser p_θ, rule F
$S \leftarrow \emptyset$, $ll \leftarrow 0$ ▷ *nothing revealed*
while masked tokens remain **do**
  $\boldsymbol{P} \leftarrow p_\theta(\cdot \mid \boldsymbol{x}^S)$ ▷ *one forward pass*
  $I \leftarrow F(\boldsymbol{x}^S)$ ▷ *same rule, same order*
  for $\ell \in I$ **do**
    $ll \mathrel{+}= \log P_\ell[x^\ell]$ ▷ *score the truth*
  $S \leftarrow S \cup I$
return ll ▷ $\log p_\theta^{\pi^F}(\boldsymbol{x})$, exact

Figure 3: **Sampling and scoring are the same loop.** Same rule, same revealed sets, same L/k forward passes — only the highlighted lines differ. P_ℓ is the denoiser's predicted distribution over the vocabulary at position ℓ, so $P_\ell[x^\ell]$ is the probability it gives the true token there.

independently (Assumption 1). Read on a fixed $\boldsymbol{x}$, the token-prediction factors are what one records by replaying generation with the true tokens revealed, at T forward passes.

Induced distribution. Summing over orders gives the distribution a policy π draws from:

$$p_\theta^\pi(\boldsymbol{x}) \;=\; \sum_\sigma p_\theta(\boldsymbol{x}, \sigma). \tag{3}$$

An MDM thus implicitly defines a family of probability distributions over text, $\{p_\theta^\pi\}_\pi$, indexed by the unmasking policy; the same trained network yields a different distribution for each policy. The sum over σ is what makes an MDM order-agnostic, and it is why its likelihood appears intractable: there are more than $L!$ ordered partitions, and each term needs its own T forward passes because the denoiser conditions on a different context under every order.

Connection to MDM training. The diffusion model of Section 2.1 is one member of this family. *Sampling:* run ancestrally, its reverse process reveals a uniformly random subset of the masked positions at each step, regardless of the tokens revealed so far. It is therefore the sampler of Equation (3) under the *uniform* policy π^{unif}, which gives every permutation probability $1/L!$ (Proposition 2), and the diffusion likelihood is $p_\theta(\boldsymbol{x}) = p_\theta^{\pi^{\mathrm{unif}}}(\boldsymbol{x})$. *Training:* the diffusion ELBO in Equation (1) equals the expected negative log-likelihood along a uniformly random order (Proposition 4; Ou et al., 2024; Zheng et al., 2024; Kim et al., 2025a),

$$-\log p_\theta^{\pi^{\mathrm{unif}}}(\boldsymbol{x}) \;\le\; \mathbb{E}_{\sigma \sim \mathrm{Unif}(\mathcal{S}_L)}\Big[\sum_{t=1}^{L} -\log p_\theta\big(x^{\sigma_t} \mid \boldsymbol{x}^{\sigma_{<t}}\big)\Big], \tag{4}$$

where the inequality is Jensen's applied to the sum in Equation (3). An MDM is thus an AO-ARM trained under uniform random ordering. *Evaluation:* the Monte Carlo estimate of Equation (4), the standard MDM metric, has the two defects of Section 1: it is a bound, and it is taken under π^{unif}, a random-order policy no large-scale MDM samples with. Section 3 removes both.

3 DUEL: Exact Likelihood under Deterministic Unmasking

In practice, leading MDMs do not sample positions at random: random ordering is reported to degrade sample quality and is not used for reported results. Instead, LLaDA, Dream, LLaDA 2.0, SDAR and Nemotron-Diffusion (Nie et al., 2025; Ye et al., 2025; Bie et al., 2025; Cheng et al., 2025; Fu et al., 2026) all generate with a deterministic *unmasking rule*: a function that reads the denoiser's predictions on the revealed tokens and returns the positions to reveal next.

Definition 2 (Unmasking rule)**.** An *unmasking rule* F maps a partially revealed sequence $\boldsymbol{x}^S$ to a non-empty set of its masked positions, $F(\boldsymbol{x}^S) \subseteq [L] \setminus S$, and is deterministic: the same input always yields the same set. It may use the denoiser's predictions $p_\theta(\cdot \mid \boldsymbol{x}^S)$, themselves a function of $\boldsymbol{x}^S$.

Instantiations. Table 4 catalogs the rules we consider, with $p_\ell^{(1)}$ and $p_\ell^{(2)}$ the highest and second-highest probabilities the denoiser assigns at masked positions ℓ, and k the number of positions revealed per step: *left-to-right* reveals the k leftmost masked positions (autoregressive when $k = 1$); *greedy confidence* (Nie et al., 2025) commits to the k most confident positions; *probability margin* (Kim et al., 2025a) prefers positions where the gap between the top two tokens is largest; *confidence threshold* (Wu et al., 2025) reveals every position with $p_\ell^{(1)} \geq \mu$ (adaptive k, falling back to the most confident position); and *KLASS* (Kim et al., 2025b) adds a stability check via the KL divergence between consecutive predictions $\hat{p}_\ell$ and p_ℓ. Each instantiation is a deterministic function of $\boldsymbol{x}^S$ (KLASS of the trajectory, via its previous prediction; Appendix C) and is thus an unmasking rule. They compose with further strategies such as block-based sampling (Arriola et al., 2025a) and entropy-bounded adaptive parallelism (Ben-Hamu et al., 2025). A pretrained denoiser with any such rule is a sampler; its determinism is what we exploit (Appendix C).

Definition 3 (DUEL sampler)**.** A DUEL *sampler* is a pair (p_θ, F) of a pretrained denoiser and an unmasking rule. The rule F induces the unmasking policy π^F as a point mass:

$$\pi^F\big(\sigma_t \mid \boldsymbol{x}^{\sigma_{<t}}\big) = \mathbf{1}\big[\sigma_t = F(\boldsymbol{x}^{\sigma_{<t}})\big]. \tag{5}$$

A DUEL sampler draws $\boldsymbol{x} \sim p_\theta^{\pi^F}$ in Equation (3) by the procedure of Algorithm 1: starting fully masked, each step reveals the positions $F(\boldsymbol{x}^S)$ with tokens drawn from $p_\theta(\cdot \mid \boldsymbol{x}^S)$, until none remain. Replaying the rule on $\boldsymbol{x}$ instead produces a single ordered partition $\sigma^F(\boldsymbol{x})$, defined recursively by $\sigma_t^F = F(\boldsymbol{x}^{\sigma^F_{<t}})$, on which π^F is a point mass: every other order must disagree with the rule at some step and receives probability zero. $\sigma^F(\boldsymbol{x})$ depends on the data, since the rule reads the revealed tokens. It is not known in advance: finding it means running the recursion, revealing $\boldsymbol{x}$ step by step.

Theorem 1 (DUEL exact likelihood)**.** *Let (p_θ, F) be a* DUEL *sampler and let $\boldsymbol{x} \in \mathcal{V}^L$ be any sequence. We can compute the* exact *likelihood of $\boldsymbol{x}$ under $p_\theta^{\pi^F}$ as:*

$$\log p_\theta^{\pi^F}(\boldsymbol{x}) \;=\; \sum_{t=1}^{T} \sum_{\ell \in \sigma_t^F} \log p_\theta\big(x^\ell \mid \boldsymbol{x}^{\sigma^F_{<t}}\big), \tag{6}$$

for the order $\sigma^F(\boldsymbol{x})$ that F produces on $\boldsymbol{x}$ and its number of steps T. Algorithm 2 implements the procedure with one forward pass per step, $T \leq L$ passes in total.

Proof sketch. Substituting the point-mass policy of Equation (5) into Equations (2) and (3),

$$p_\theta^{\pi^F}(\boldsymbol{x}) = \sum_\sigma \prod_t \mathbf{1}\big[\sigma_t = F(\boldsymbol{x}^{\sigma_{<t}})\big] \prod_{\ell \in \sigma_t} p_\theta\big(x^\ell \mid \boldsymbol{x}^{\sigma_{<t}}\big) = \prod_t \prod_{\ell \in \sigma_t^F} p_\theta\big(x^\ell \mid \boldsymbol{x}^{\sigma^F_{<t}}\big),$$

since the indicators vanish for every order except $\sigma^F(\boldsymbol{x})$. Full proof in Appendix B.4. □

The product on the right is the chain rule along $\sigma^F(\boldsymbol{x})$: the order-aware computation of a model with one *fixed*, known order (e.g. AR), applied to an MDM, an order-agnostic model trained on *every* order. The novel observation is the second equality. The order-agnostic likelihood in Equation (3) is a sum over more than $L!$ orders, each needing its own forward passes, yet a deterministic rule zeroes all but one term. Replaying the rule on $\boldsymbol{x}$ finds that term without knowing the order in advance. DUEL has no Jensen gap and evaluates the sampler actually used; the ELBO in Equation (4) has a Jensen gap and evaluates a random-order policy no one uses. Computing the likelihood costs exactly what generation costs: $\lceil L/k \rceil$ forward passes for k tokens unmasked in parallel per step. Since this grows with length, DUEL is cheaper than the ELBO's MC-sample Monte Carlo estimate on short spans and costlier on long ones; whether that is worth it depends on the use case (Appendix E.1).

4 Experiments

Scoring MDMs with DUEL, not the ELBO, changes our conclusions. We show this in four likelihood-based evaluations: perplexity, scaling laws, multiple-choice benchmarks, and sampler selection.

Table 1: **Held-out and zero-shot perplexity.** Gap closed (%) is the share of the ELBO's MDM–AR perplexity gap that DUEL removes; ELBO bound under π^{unif}, DUEL exact under π^{F}, **bold** wins.

(a) Held-out OWT perplexity. AR baseline: 17.54.

Model	ELBO↓	DUEL↓	Gap Closed↑
SEDD	≤24.10	**22.58**	23.2
MDLM	≤22.98	**21.86**	20.6
BD3-LM L'=4	≤20.73	**19.73**	31.3
BD3-LM L'=8	≤21.68	**20.37**	31.6
BD3-LM L'=16	≤22.27	**20.76**	31.9

(b) Held-out LM1B perplexity. AR baseline: 22.83.

Model	ELBO↓	DUEL↓	Gap Closed↑
SEDD	≤32.68	**31.51**	11.9
MDLM	≤31.78	**29.03**	30.7
BD3-LM L'=4	≤28.23	**26.97**	23.3
BD3-LM L'=8	≤29.83	**27.85**	28.3
BD3-LM L'=16	≤30.60	**28.39**	28.4

(c) Zero-shot gap closed (%, ↑); trained on OWT, BD3-LM L'=4. Full PPL in Table 5.

Dataset	SEDD	MDLM	BD3-LM
PTB	31.3	34.3	81.8
Wikitext	40.8	28.5	31.4
LM1B	22.1	29.0	29.8
AG News	25.7	27.8	51.7
Average	30.0	29.9	48.7

(d) Zero-shot perplexity (↓), 8B models.

Model	Estimator	Wikitext	Lambada	AG News
Llama3	exact	7.94	32.40	41.29
LLaDA	ELBO	≤15.29	≤39.04	≤85.17
	DUEL	**14.50**	**36.00**	**78.91**

4.1 Perplexity

Setup. MDMs have lagged behind AR models in perplexity benchmarks (Nie et al., 2024). We ask whether this gap is inflated by evaluation methodology. We compare an AR baseline, SEDD (Lou et al., 2024), MDLM (Sahoo et al., 2024), and BD3-LM (Arriola et al., 2025a) with block sizes $L' \in \{4, 8, 16\}$, all with comparable sizes, compute, and data. All MDMs use block-greedy-confidence unmasking at $k = 1$: SEDD and MDLM with $L' = 4$, BD3-LM at its native L', LLaDA at $L' = 32$. We evaluate *in-domain* on OpenWebText (OWT) and LM1B (Tables 1a and 1b), *zero-shot* on PTB, Wikitext, LM1B, Lambada, and AG News (Table 1c, full in Table 5), and at scale with LLaDA-8B (Nie et al., 2025) and Llama3-8B (Grattafiori et al., 2024) on Wikitext, Lambada, and AG News (Table 1d). Full details in Appendix D.1.

Metric. The *perplexity gap* is $\Delta = \mathrm{PPL}_{\mathrm{MDM}} - \mathrm{PPL}_{\mathrm{AR}}$. Δ_{ELBO} inflates it on two axes—a loose bound and the wrong policy (π^{unif} instead of π^{F})—both of which Δ_{DUEL} corrects. The *gap closed* is the share of Δ_{ELBO} this eliminates, $(\Delta_{\mathrm{ELBO}} - \Delta_{\mathrm{DUEL}})/\Delta_{\mathrm{ELBO}}$, in percent.

Results. DUEL **closes 12–32% of the MDM–AR gap in-domain and 22–82% zero-shot**, on every model and corpus (Tables 1a to 1c; full perplexities in Table 5). At 8B scale (Table 1d), DUEL lowers LLaDA-8B perplexity by 5–8% on Wikitext, Lambada and AG News; since LLaDA and Llama3 differ in training data and compute, we report raw perplexity instead of the gap closed. Proper perplexity evaluation therefore *substantially* closes the MDM–AR perplexity gap, and **the gain comes entirely from evaluation, not from any change to the models**: the ELBO averages over all orderings equally, including poor ones, while DUEL scores the same checkpoints under the deterministic policy they are actually run with.

4.2 Scaling laws

Setup. Nie et al. (2024) conclude that an MDM needs $\sim 16\times$ the training compute of an AR model to reach the same validation loss for their SMDM models. That comparison scores the AR side with an exact likelihood and the MDM side with the ELBO—a bound, under the training policy π^{unif}, on a distribution nobody samples from. We redo it with the exact DUEL likelihood and change nothing else. Their released AR and MDM IsoFLOP grids are taken unchanged: non-embedding sizes 19M–1028M at budgets $C \in \{6, 10, 30, 60, 100\} \times 10^{18}$ FLOPs, under their convention $C = 6ND$. At every checkpoint we recompute three losses ourselves, on the same 800 held-out SlimPajama sequences: the exact AR likelihood; the MDM ELBO with $\mathrm{MC} = 128$ Monte Carlo samples, the Nie et al. (2024) default; and DUEL exact at $k = 1$ under *left-to-right* unmasking—the order closest

Table 2: **The MDM–AR compute gap across SMDM's scaling law.** Compute-optimal frontier on their IsoFLOP grid; loss in nats/token, gap the equal-loss multiplier over AR. Both MDM blocks are the same checkpoints: the ELBO a bound under π^{unif}, DUEL exact under π^{F} (left-to-right). Slopes a in $L_{\text{opt}} \propto C^{-a}$: AR 0.0645, ELBO 0.0626, DUEL 0.0661 (Table 6).

	AR	MDM ELBO		MDM DUEL		
Compute (FLOPs)	**loss**	**loss**	**gap**	**loss**	**gap**	**Gap cut**
6×10^{18}	2.918	3.449	$\times$14.6	3.398	$\times$10.0	$\mathbf{-31\%}$
10×10^{18}	2.818	3.339	$\times$14.8	3.279	$\times$9.9	$\mathbf{-33\%}$
30×10^{18}	2.620	3.110	$\times$15.3	3.044	$\times$9.6	$\mathbf{-37\%}$
60×10^{18}	2.508	2.977	$\times$15.7	2.908	$\times$9.5	$\mathbf{-40\%}$
100×10^{18}	2.436	2.897	$\times$15.9	2.825	$\times$9.4	$\mathbf{-41\%}$

to AR generation, and so the most comparable to the AR side. ELBO and DUEL score identical sequences, so their difference is paired and more precise than either loss. Full details in Appendix D.2.

Metric. The fitting procedure is SMDM's (Nie et al., 2024), with only the y-axis changed. Per budget we fit a quadratic of loss against $\log N$ and take its vertex as the compute-optimal $(N_{\text{opt}}, L_{\text{opt}})$; we then fit $L_{\text{opt}} \propto C^{-a}$ across budgets by OLS in log-log space. The *discrete* best checkpoint also agrees between the ELBO and DUEL at four of the five budgets (Table 6). The equal-loss compute gap at budget C solves $L_{\text{MDM}}(x) = L_{\text{AR}}(C)$ and reports x/C, inverting each frontier with *its own* fitted exponent a (Appendix D.2). This is the horizontal distance the arrows in Figure 1 draw.

Results. Table 2 gives the scaling law frontier, visualized in Figure 1. Our ELBO frontier reproduces the published $\sim$16$\times$ MDM–AR compute gap, confirming we redo Nie et al.'s analysis and not a different one. The exact DUEL likelihood then **drops the compute gap from $\times$15–16 to $\times$9–10, cutting roughly 30–40% of it at every budget**. The cut *grows* with scale: -31% at 6×10^{18} FLOPs, -41% at 10^{20}. The two axes are related exponentially, so the ELBO's $\approx$0.07-nat slack, negligible on a loss curve, produces the whole discrepancy. Whether masked diffusion is competitive with autoregression at scale is judged largely on the MDM–AR compute gap multiplier. Scoring AR exactly and MDM with the ELBO is not a neutral comparison; it is biased against MDM. Scaling laws therefore need the likelihood *exact*, not merely tight—which for MDMs is what DUEL provides.

4.3 Multiple-choice benchmarks

Setup. A multiple-choice benchmark scores each candidate answer by its likelihood and returns the highest, so the estimator is part of the predictor. We evaluate LLaDA-8B-Base on six such tasks, scored three ways: the ELBO at $\text{MC} = 128$ (the LLaDA default, a bound under the training policy π^{unif}), and DUEL at $k = 1$ under greedy-confidence and left-to-right unmasking, both exact under their own π^{F}. We keep only tasks whose answer choices average roughly 20 tokens or more, since shorter spans leave almost no ordering for the ELBO to get wrong. Full details in Appendix D.3.

Accuracy. We score with accuracy, length-normalized on HellaSwag, following the original LLaDA evaluation (Nie et al., 2025), for all three scorers. **DUEL improves accuracy on all six tasks with no change to the model: $+2.5$ points on average under greedy sampling and $+4.2$ under left-to-right sampling** (Table 3a). The single largest gain is $+9.7$ points, on LSAT-RC with left-to-right sampling.

Calibration. **DUEL gives a better distribution over answers, not merely a better argmax.** Per-choice log-likelihoods are token log-probability sums, so a softmax over them is wildly overconfident and the scorers differ in scale as well as ranking. Following Guo et al. (2017), we temperature-scale per task and scorer by 2-fold cross-fitting and report Brier, NLL, and AUROC (Table 3b). Both DUEL variants beat the ELBO on every task by Brier and NLL and on five of six by AUROC.

Table 3: **Multiple-choice accuracy and calibration under the ELBO and DUEL.** LLaDA-8B-Base on six tasks, scored three ways. Both DUEL columns beat the ELBO in every row of (a) and (b).

(a) Accuracy (%, ↑); gain over the ELBO in parentheses.

	ELBO	DUEL	
Task	**uniform**	**greedy**	**left-to-right**
MuTual	50.1	51.6 (+1.5)	51.7 (+1.6)
LogiQA	34.1	39.0 (+4.9)	41.6 (+7.5)
LSAT-RC	55.4	60.2 (+4.8)	65.1 (+9.7)
PIQA	74.8	76.1 (+1.3)	76.7 (+1.9)
LSAT-LR	41.0	42.5 (+1.5)	45.0 (+4.0)
HellaSwag	70.7	71.4 (+0.7)	71.1 (+0.4)
Average	54.3	56.8 (+2.5)	58.5 (+4.2)

(b) Calibration, six-task average. Per task in Table 7.

	ELBO	DUEL	
Metric	**uniform**	**greedy**	**left-to-right**
Brier ↓	0.627	0.608	0.594
NLL ↓	1.170	1.148	1.125
AUROC ↑	0.597	0.630	0.644

4.4 Sampler selection

Setup. An MDM's performance greatly depends on the sampler used. Benchmarks can sometimes select samplers, but not always: small research models score near chance, and they require fixing target tasks in advance. A likelihood is a good alternative, but the ELBO is invariant to the unmasking rule F—it scores under π^{unif} whatever rule is run—so it cannot rank samplers. DUEL scores each policy under π^F itself, so it can rank them; we test whether that ranking is right. We generate at two scales: BD3-LM (Arriola et al., 2025a) (110M, OWT, unconditional) and LLaDA-8B-Base (Nie et al., 2025) (prefix-conditioned). We compare left-to-right, greedy confidence and probability margin, all under π^F, at matched budgets: revealing $k \in \{1, 2, 4, 8\}$ tokens per pass gives four NFE levels. (Confidence threshold is excluded: its NFE varies.) Full details in Appendix D.4.

Metric. We rank the samplers by DUEL perplexity and by two standard sampler-sensitive quantities, generative perplexity (gen-ppl) and MAUVE (Pillutla et al., 2021), and take an LLM judge as ground truth. The three read different text: DUEL scores *held-out* OWT under the policy — never the model's own samples — gen-ppl scores generated text under GPT-2 Large (Radford et al., 2019), and MAUVE compares that text to an OWT reference. The judge, Claude Sonnet 5, is shown one generation from each of two samplers at the same budget—blinded, in both A/B orders—and asked which is higher quality, weighing fluency, grammaticality, coherence and consistency; it returns a winner and the dimension that drove it. Each pair aggregates $n \approx 500$ comparisons and counts as *decided* when its win-rate CI excludes 0.5 (Table 10).

Results. DUEL agrees with the LLM judge on all 14 decided pairs, against 8/14 for gen-ppl and 12/14 for MAUVE (Table 10). Adding the 10 undecided pairs, scored by point direction, DUEL leads 21/24 against 15/24 and 16/24. All the discrimination is at 110M, where gen-ppl agrees on 1/7; at 8B all three agree 7/7. DUEL perplexity tracks textual quality itself, not a proxy: it correlates with *every* dimension the judge scores—fluency, grammaticality, coherence and consistency ($\rho = -0.92$ to -0.99)—and the verdicts it agrees with are decided by *coherence* (79% at 110M, 55% at 8B), not by surface grammar (Appendix D.4).

Why gen-ppl fails. Both DUEL and gen-ppl are cross-entropies, so each decomposes into an entropy and KL divergence; the two differ only in which distribution fills each slot. gen-ppl scores the model's own samples with an external scorer q, while DUEL scores held-out data with the model. Writing p_{data} for held-out OWT and $p_\theta^{\pi^F}$ for the text it induces under the sampler π^F, per token:

$$\log \text{gen-ppl}(F) = H(p_\theta^{\pi^F}) + D_{\text{KL}}(p_\theta^{\pi^F} \| q), \quad \log \text{DUEL-ppl}(F) = H(p_{\text{data}}) + D_{\text{KL}}(p_{\text{data}} \| p_\theta^{\pi^F}). \tag{7}$$

In gen-ppl, the entropy $H(p_\theta^{\pi^F})$ is the *sampler's own*, so a policy can lower its score by collapsing onto repetitive, low-entropy text without getting closer to real data (Pynadath et al., 2026). In DUEL the entropy is $H(p_{\text{data}})$, a constant, so only the KL moves. gen-ppl therefore rewards entropy collapse while DUEL penalizes it. Thus a sampler can win on gen-ppl by producing *worse* text, but it can win

on DUEL only by producing text closer to the true data distribution. We see exactly this at 110M: at every low-NFE budget the left-to-right sampler produces the most repetitive text (token entropy 4.0–4.6, against ≈4.9 for its peers), and gen-ppl ranks it *best* while the LLM judge ranks it *worst*.

5 Discussion

Summary. An MDM is a family of distributions indexed by unmasking policy. The ELBO bounds the uniform-policy member no deployed sampler uses; DUEL scores the member actually sampled, exactly. The picture changes: under their samplers, MDMs are far closer to AR models than the ELBO suggests, in perplexity, scaling, benchmarks, and sampler selection.

Related work. Existing routes to likelihood or non-uniform orderings in MDMs modify training or architecture. Planner-aware training (PAPL (Peng et al., 2025b), PUMA (Kim et al., 2026)) and learned orders (Peng et al., 2025a; Wang et al., 2025b) realign training with a non-uniform policy, at the cost of non-trivial changes to the training procedure, and still evaluate through a variational bound whose Jensen gap remains. Asymptotically exact likelihood has required Eso-LMs' causal attention with an explicit AO-ARM factorization (Sahoo et al., 2025) or continuous-time optimal-denoiser assumptions (Jeon et al., 2025); Chen et al. (2025) use exact per-trajectory likelihood for GRPO training. DUEL is complementary and plug-and-play: exact likelihood for any standard MDM under any deterministic policy, post hoc, with no training or architectural change. See Appendix A.

Limitations. DUEL covers deterministic position selection. Token-level heuristics such as nucleus or top-k sampling are invisible to it, as they are to autoregressive perplexity. Stochastic policies and remasking samplers, which revisit revealed tokens, break the collapse in Theorem 1: the policy is no longer a point mass and the sum over orders returns. Its cost exceeds the ELBO's on long sequences, worthwhile for final evaluations but not for training monitoring (Appendix E.1).

AI use statement

In this work, we used generative AI tools (LLM coding assistants) to implement methods, to help design experiments and provide feedback on the methodology, to interpret results, and to help formulate mathematical claims and write proofs. We have not used generative AI tools to generate synthetic data sets, develop theoretical frameworks, or propose hypotheses, and translation, dataset cleaning, and qualitative data analysis are not applicable to this work. Additionally, we used generative AI tools to create and edit software code, figures and tables, to draft parts of the paper and edit it for readability, and to search for, summarize, and format references to related literature. Separately from this assistance, an LLM is part of the experimental design: Claude Sonnet 5 is the judge in the sampler-selection study (Section 4.4), with prompts, blinding, and pairing protocol given in Appendix D.4. We have reviewed all AI-assisted work: the authors checked every AI-assisted claim and proof, and verified AI-written code and the numbers it produced against logged runs. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Ethics statement

This work proposes an evaluation method for an existing model class. It evaluates public datasets and publicly released checkpoints, trains no models, collects no new data, involves no human subjects, and releases no models or data with misuse risk. All datasets and models are cited and used under their public licenses for research. We do not foresee specific societal harms from more accurate likelihood evaluation of masked diffusion models.

Reproducibility statement

DUEL requires no training: every checkpoint is a public release evaluated unchanged, listed with its training budget and evaluated block size in Appendix F. The main result, Theorem 1 (Section 3), is proved in full in Appendix B.4, with its assumption and the supporting results in Appendix B. Appendix D specifies each experiment's protocol (unmasking rules, Monte Carlo budgets, sequence

lengths, the IsoFLOP fitting procedure, and the judge prompts and pairing protocol), Appendix E.1 reports measured costs, and Appendix F gives hardware and numerics. Code for DUEL scoring and all evaluations will be released upon acceptance.

REFERENCES

Marianne Arriola and Volodymyr Kuleshov. Set diffusion: Interpolating token orderings between autoregression and diffusion for fast and flexible decoding. In *International Conference on Machine Learning*, 2026. arXiv:2607.01775.

Marianne Arriola, Aaron Gokaslan, Justin T Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Block diffusion: Interpolating between autoregressive and diffusion language models. *arXiv preprint arXiv:2503.09573*, 2025a.

Marianne Arriola, Yair Schiff, Hao Phung, Aaron Gokaslan, and Volodymyr Kuleshov. Encoder-decoder diffusion language models for efficient training and inference. *arXiv preprint arXiv:2510.22852*, 2025b.

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.

Simone Balloccu, Patrícia Schmidtová, Mateusz Lango, and Ondřej Dušek. Leak, cheat, repeat: Data contamination and evaluation malpractices in closed-source llms. *arXiv preprint arXiv:2402.03927*, 2024.

Heli Ben-Hamu, Itai Gat, Daniel Severo, Niklas Nolte, and Brian Karrer. Accelerated sampling from masked diffusion models via entropy bounded unmasking. *arXiv preprint arXiv:2505.24857*, 2025.

Tiwei Bie, Maosong Cao, Kun Chen, Lun Du, Mingliang Gong, Zhuochen Gong, Yanmei Gu, Jiaqi Hu, Zenan Huang, Zhenzhong Lan, et al. LLaDA 2.0: Scaling up diffusion language models to 100b. *arXiv preprint arXiv:2512.15745*, 2025.

Yuri Burda, Roger B Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. In *International Conference on Learning Representations*, 2016.

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.

Shirui Chen, Jiantao Jiao, Lillian J Ratliff, and Banghua Zhu. dUltra: Ultra-fast diffusion language models via reinforcement learning. *arXiv preprint arXiv:2512.21446*, 2025.

Shuang Cheng, Yihan Bian, Dawei Liu, Linfeng Zhang, Qian Yao, Zhongbo Tian, Wenhai Wang, Qipeng Guo, Kai Chen, Biqing Qi, and Bowen Zhou. SDAR: A synergistic diffusion-autoregression paradigm for scalable sequence generation. *arXiv preprint arXiv:2510.06303*, 2025.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. A discourse-aware attention model for abstractive summarization of long documents. *arXiv preprint arXiv:1804.05685*, 2018.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186, 2019.

Yonggan Fu, Lexington Whalen, Abhinav Garg, Chengyue Wu, Maksim Khadkevich, Nicolai Oswald, Enze Xie, Daniel Egert, Sharath Turuvekere Sreenivas, Shizhe Diao, Chenhan Yu, Ye Yu, Weijia Chen, Sajad Norouzi, Jingyu Liu, Shiyi Lan, Ligeng Zhu, Jin Wang, Jindong Jiang, Morteza Mardani, Mehran Maghoumi, Song Han, Ante Jukić, Nima Tajbakhsh, Jan Kautz, and Pavlo Molchanov. Nemotron-Labs-Diffusion: A tri-mode language model unifying autoregressive, diffusion, and self-speculation decoding. *arXiv preprint arXiv:2607.05722*, 2026.

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

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pp. 1321–1330. PMLR, 2017.

Satoshi Hayakawa, Yuhta Takida, Masaaki Imaizumi, Hiromi Wakaki, and Yuki Mitsufuji. Demystifying maskgit sampler and beyond: Adaptive order selection in masked diffusion. *arXiv preprint arXiv:2510.04525*, 2025.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Emiel Hoogeboom, Alexey A Gritsenko, Jasmijn Bastings, Ben Poole, Rianne van den Berg, and Tim Salimans. Autoregressive diffusion models. *arXiv preprint arXiv:2110.02037*, 2021a.

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, 2021b.

Zemin Huang, Yuhang Wang, Zhiyang Chen, and Guo-Jun Qi. Don't settle too early: Self-reflective remasking for diffusion language models. *arXiv preprint arXiv:2509.23653*, 2025.

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.

Metod Jazbec, Theo X Olausson, Louis Béthune, Pierre Ablin, Michael Kirchhof, Joao Monterio, Victor Turrisi, Jason Ramapuram, and Marco Cuturi. Learning unmasking policies for diffusion language models. *arXiv preprint arXiv:2512.09106*, 2025.

Moongyu Jeon, Sangwoo Shin, Dongjae Jeon, and Albert No. Information-theoretic discrete diffusion. *arXiv preprint arXiv:2510.24088*, 2025.

Jaeyeon Kim, Kulin Shah, Vasilis Kontonis, Sham Kakade, and Sitan Chen. Train for the worst, plan for the best: Understanding token ordering in masked diffusions. *arXiv preprint arXiv:2502.06768*, 2025a.

Jaeyeon Kim, Jonathan Geuter, David Alvarez-Melis, Sham Kakade, and Sitan Chen. Stop training for the worst: Progressive unmasking accelerates masked diffusion training. *arXiv preprint arXiv:2602.10314*, 2026.

Seo Hyun Kim, Sunwoo Hong, Hojung Jung, Youngrok Park, and Se-Young Yun. KLASS: KL-guided fast inference in masked diffusion models. *arXiv preprint arXiv:2511.05664*, 2025b.

Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

Hugo Larochelle and Iain Murray. The neural autoregressive distribution estimator. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 29–37. JMLR Workshop and Conference Proceedings, 2011.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 32819–32848, 2024.

Omer Luxembourg, Haim Permuter, and Eliya Nachmani. Plan for speed–dilated scheduling for masked diffusion language models. *arXiv preprint arXiv:2506.19037*, 2025.

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.

Shen Nie, Fengqi Zhu, Chao Du, Tianyu Pang, Qian Liu, Guangtao Zeng, Min Lin, and Chongxuan Li. Scaling up masked diffusion models on text. *arXiv preprint arXiv:2410.18514*, 2024.

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.

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.

Fred Zhangzhi Peng, Zachary Bezemek, Sawan Patel, Jarrid Rector-Brooks, Sherwood Yao, Avishek Joey Bose, Alexander Tong, and Pranam Chatterjee. Path planning for masked diffusion model sampling. *arXiv preprint arXiv:2502.03540*, 2025a.

Fred Zhangzhi Peng, Zachary Bezemek, Jarrid Rector-Brooks, Shuibai Zhang, Anru R Zhang, Michael Bronstein, Avishek Joey Bose, and Alexander Tong. Planner aware path learning in diffusion language models training. *arXiv preprint arXiv:2509.23405*, 2025b.

Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. Mauve: Measuring the gap between neural text and human text using divergence frontiers. *Advances in Neural Information Processing Systems*, 34:4816–4828, 2021.

Patrick Pynadath, Jiaxin Shi, and Ruqi Zhang. Generative frontiers: Why evaluation matters for diffusion language models, 2026. URL https://arxiv.org/abs/2604.02718.

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.

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

Subham Sekhar Sahoo, Zhihan Yang, Yash Akhauri, Johnna Liu, Deepansha Singh, Zhoujun Cheng, Zhengzhong Liu, Eric Xing, John Thickstun, and Arash Vahdat. Esoteric language models. *arXiv preprint arXiv:2506.01928*, 2025.

Yair Schiff, Subham Sekhar Sahoo, Hao Phung, Guanghan Wang, Sam Boshar, Hugo Dalla-torre, Bernardo P de Almeida, Alexander Rush, Thomas Pierrot, and Volodymyr Kuleshov. Simple guidance mechanisms for discrete diffusion models. *arXiv preprint arXiv:2412.10193*, 2024.

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.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. pmlr, 2015.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

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.

Benigno Uria, Iain Murray, and Hugo Larochelle. A deep and tractable density estimator. In *International Conference on Machine Learning*, pp. 467–475. PMLR, 2014.

Benigno Uria, Marc-Alexandre Côté, Karol Gregor, Iain Murray, and Hugo Larochelle. Neural autoregressive distribution estimation. *Journal of Machine Learning Research*, 17(205):1–37, 2016.

Guanghan Wang, Yair Schiff, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Remasking discrete diffusion models with inference-time scaling. *arXiv preprint arXiv:2503.00307*, 2025a.

Zhe Wang, Jiaxin Shi, Nicolas Heess, Arthur Gretton, and Michalis K Titsias. Learning-order autoregressive models with application to molecular graph generation. *arXiv preprint arXiv:2503.05979*, 2025b.

Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Ben Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Siddartha Naidu, et al. Livebench: A challenging, contamination-free llm benchmark. *arXiv preprint arXiv:2406.19314*, 4, 2024.

Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. Fast-dllm: Training-free acceleration of diffusion llm by enabling kv cache and parallel decoding. *arXiv preprint arXiv:2505.22618*, 2025.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. XLNet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7B: Diffusion large language models. *arXiv preprint arXiv:2508.15487*, 2025.

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. *arXiv preprint arXiv:2409.02908*, 2024.

CONTENTS

1 Introduction **1**

2 Preliminaries **2**

2.1 Masked Diffusion Models 2

2.2 Masked Diffusion as Any-Order Autoregression 3

3 DUEL: Exact Likelihood under Deterministic Unmasking **4**

4 Experiments **5**

4.1 Perplexity 6

4.2 Scaling laws 6

4.3 Multiple-choice benchmarks 7

4.4 Sampler selection 8

5 Discussion **9**

A Related work **15**

B Proofs **17**

B.1 The joint factorization 17

B.2 Ancestral sampling realizes the uniform policy 18

B.3 The ELBO is the uniform-order NLL 18

B.4 Proof of the exact-likelihood theorem 19

C Unmasking rules **21**

D Experiments **23**

D.1 Perplexity 23

D.2 Scaling laws 25

D.3 Multiple-choice benchmarks 26

D.4 Sampler selection 27

E Additional experiments **33**

E.1 Cost 33

E.2 Estimator decomposition 34

E.3 Sampler selection and downstream accuracy 35

F Reproducibility **38**

A Related work

Masked diffusion. Diffusion was developed for continuous data (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2020) and carried to discrete state spaces (Hoogeboom et al., 2021b; Austin et al., 2021; Campbell et al., 2022); Lou et al. (2024) replaced the variational objective with score entropy. The absorbing (masking) kernel proved the most effective special case, and Sahoo et al. (2024) and Shi et al. (2024) reduced its objective to a weighted masked-language-modeling loss. Zheng et al. (2024) showed the time embedding is redundant, which is what lets the denoiser be written $p_\theta(\boldsymbol{x}^\ell \mid \boldsymbol{x}^S)$ throughout this paper. MDMs now run at scale: LLaDA (Nie et al., 2025), Dream (Ye et al., 2025), LLaDA-2.0 (Bie et al., 2025), SDAR (Cheng et al., 2025), Nemotron-Diffusion (Fu et al., 2026), and fast commercial systems (Inception Labs et al., 2025; Song et al., 2025). Block diffusion (Arriola et al., 2025a) and E2D2 (Arriola et al., 2025b) interpolate toward autoregression to recover KV caching. SMDM (Nie et al., 2024) fits the compute-optimal scaling law we re-run in Section 4.2.

Any-order autoregression. Order-agnostic density estimation predates diffusion. NADE (Larochelle & Murray, 2011) fixed one order; its order-agnostic successors (Uria et al., 2014; 2016) train one network on orders drawn uniformly at random; XLNet (Yang et al., 2019) carried permutation language modeling into pretraining. ARDMs (Hoogeboom et al., 2021a) made the link to diffusion explicit, training an any-order autoregressive model with a diffusion-style objective. Ou et al. (2024) and Zheng et al. (2024) closed the loop in the other direction: the absorbing-diffusion bound *is* the any-order autoregressive objective, so a trained MDM already is an any-order autoregressive model. Propositions 3 and 4 are the form of that equivalence we use. Kim et al. (2025a) study its cost — the model must solve every infilling problem (S, ℓ) rather than the L prefix problems, and the hardest of them dominate — and train one causal network per fixed order to measure it. LO-ARM (Wang et al., 2025b) learns the order rather than fixing it, training an order-policy jointly with the denoiser, with results on molecular graphs.

Likelihood in masked diffusion. Bounds come first: Sahoo et al. (2024) and Shi et al. (2024) derived the objective of Equation (1), Ou et al. (2024) identified it with the any-order autoregressive likelihood, and all three bound the π^{unif} marginal. Jeon et al. (2025) make cross-entropy exact through an information-theoretic identity, but only for the optimal denoiser in continuous time, which a trained finite-step model is not. Other work changes training. PAPL (Peng et al., 2025b) trains under a planner-aware ELBO whose exact form requires simulating the planner, so its practical loss is a reweighting of the standard one, and it reports no likelihood values; PUMA (Kim et al., 2026) modifies the forward masking process so that training masks resemble a test-time rule's, at no extra cost. Both remain bounds on a non-uniform member, and a checkpoint from either is scored by DUEL unchanged. Sampling-time planners (Peng et al., 2025a) and learned orders (Wang et al., 2025b) are unmasking policies in our sense. Eso-LMs (Sahoo et al., 2025) obtain an asymptotically exact, importance-weighted likelihood by changing the architecture, and Chen et al. (2025) use an exact per-trajectory likelihood as the log-probability term in RL post-training, the same quantity DUEL computes, put to a different use. A fixed-order likelihood does not require fixed-order training: the σ-learners of Kim et al. (2025a) train one network per order, but order-agnostic training (Uria et al., 2014) exists precisely so that one network can be evaluated along any order. What Uria et al. (2014) and the importance-weighted estimator of Burda et al. (2016) report, however, are averages over orders drawn independently of $\boldsymbol{x}$, i.e. quantities of the π^{unif} member; the exhaustive limit is the order-marginal we enumerate in Appendix E.2. DUEL evaluates π^F instead: the order is a function of $\boldsymbol{x}$ through the model's own predictions, differs for every input, and is known only once F has been replayed.

Unmasking policies and samplers. Samplers differ in which positions they reveal. The position-selection rules, learned policies and reveal schedules in the literature are all deterministic given the revealed tokens, so DUEL scores every one of them; Appendix C lists them. Set diffusion (Arriola & Kuleshov, 2026) instead fixes an order distribution in advance and trains under it, which tightens its bound but leaves it a bound; a set-diffusion checkpoint sampled with a deterministic rule is scored exactly by Theorem 1. Remasking (Wang et al., 2025a; Huang et al., 2025) revisits revealed positions, breaks the ordered-partition structure behind Proposition 1, and is out of scope.

Evaluating MDMs. Each use of an exact likelihood in this paper has a substitute with known failure modes. Generative perplexity scores samples under an external model, usually GPT-2 (Radford et al., 2019), and rewards low-entropy output because its entropy term is the sampler's own (Pynadath et al., 2026), as our judge study shows (Section 4.4); token entropy and MAUVE (Pillutla et al., 2021) only partly correct this. Benchmark accuracy is exposed to contamination (Balloccu et al., 2024) and gaming (White et al., 2024), and for likelihood-scored multiple choice depends on the estimator itself (Section 4.3). For MDMs the likelihood was unavailable, the ELBO being a bound on the wrong member of the family; DUEL supplies it, exact and under the sampler actually used, on the same footing as an AR model's perplexity.

B PROOFS

Throughout, $\boldsymbol{x} \in \mathcal{V}^L$ is a clean sequence, $S \subseteq [L]$ a set of revealed positions, and $\boldsymbol{x}^S$ the partial sequence carrying the tokens of $\boldsymbol{x}$ at those positions together with their position indices; $\boldsymbol{x}^S$ therefore determines S. An order $\sigma = (\sigma_1, \ldots, \sigma_T)$ is an ordered partition of $[L]$, with $\sigma_{<t} = \sigma_1 \cup \cdots \cup \sigma_{t-1}$; the *sequential* case is $|\sigma_t| = 1$ for all t, so $T = L$ and $\sigma \in \mathcal{S}_L$.

B.1 THE JOINT FACTORIZATION

Generation alternates position selection and token prediction. An *unmasking policy* $\pi(\sigma_t \mid \boldsymbol{x}^{\sigma_{<t}})$ is a conditional distribution over the next set of positions that reads the past only through the revealed tokens. The tokens at the selected positions are then drawn from the denoiser.

Assumption 1 (Within-step independence)**.** At each step the sampler draws the tokens at the selected positions independently, from the denoiser's per-position marginals:

$$p_\theta\big(\boldsymbol{x}^{\sigma_t} \mid \boldsymbol{x}^{\sigma_{<t}}\big) \;=\; \prod_{\ell \in \sigma_t} p_\theta\big(\boldsymbol{x}^{\ell} \mid \boldsymbol{x}^{\sigma_{<t}}\big).$$

Assumption 1 is vacuous in the sequential case, where $|\sigma_t| = 1$. For $|\sigma_t| > 1$ it is not an approximation but a description of the sampler: the denoiser emits one categorical distribution per position and Algorithm 1 draws from them independently, so the assumption holds by construction for the model's own distribution p_θ^π. It is an approximation only if one asks whether p_θ^π matches a data distribution with within-step dependencies.

Proposition 1 (Joint factorization)**.** *Let π be an unmasking policy and let σ range over ordered partitions of $[L]$ into T parts. The generative process defines a joint distribution over $(\boldsymbol{x}, \sigma)$,*

$$p_\theta(\boldsymbol{x}, \sigma) \;=\; \prod_{t=1}^{T} \pi\big(\sigma_t \mid \boldsymbol{x}^{\sigma_{<t}}\big) \cdot p_\theta\big(\boldsymbol{x}^{\sigma_t} \mid \boldsymbol{x}^{\sigma_{<t}}\big), \qquad p_\theta^\pi(\boldsymbol{x}) \;=\; \sum_{\sigma} p_\theta(\boldsymbol{x}, \sigma), \tag{8}$$

and under Assumption 1 the second factor splits over $\ell \in \sigma_t$. In the sequential case $T = L$ and Equation (8) is Equation (2).

Proof. A run of the sampler is the interleaved sequence $(\sigma_1, \boldsymbol{x}^{\sigma_1}, \sigma_2, \boldsymbol{x}^{\sigma_2}, \ldots, \sigma_T, \boldsymbol{x}^{\sigma_T})$, which determines and is determined by the pair $(\boldsymbol{x}, \sigma)$. Applying the chain rule to it and splitting each step into its two draws,

$$\begin{aligned} p_\theta(\boldsymbol{x}, \sigma) &= \prod_{t=1}^{T} p_\theta\big(\boldsymbol{x}^{\sigma_t}, \sigma_t \mid \boldsymbol{x}^{\sigma_{<t}}, \sigma_{<t}\big) \\ &= \prod_{t=1}^{T} \pi\big(\sigma_t \mid \boldsymbol{x}^{\sigma_{<t}}, \sigma_{<t}\big) \cdot p_\theta\big(\boldsymbol{x}^{\sigma_t} \mid \sigma_t, \boldsymbol{x}^{\sigma_{<t}}, \sigma_{<t}\big) \\ &= \prod_{t=1}^{T} \pi\big(\sigma_t \mid \boldsymbol{x}^{\sigma_{<t}}\big) \cdot p_\theta\big(\boldsymbol{x}^{\sigma_t} \mid \boldsymbol{x}^{\sigma_{<t}}\big). \end{aligned}$$

The first line is the chain rule, with $\boldsymbol{x}^{\sigma_{<t}} = \emptyset$ and $\sigma_{<t} = \emptyset$ at $t = 1$. The second applies $p(A, B \mid C) = p(B \mid C)\, p(A \mid B, C)$ and names the first factor the policy. In the third, $\sigma_{<t}$ is dropped because $\boldsymbol{x}^{\sigma_{<t}}$ carries position indices and hence determines $\sigma_{<t}$; and σ_t is dropped from the second factor because the denoiser's input is the partial sequence $\boldsymbol{x}^{\sigma_{<t}}$ alone, so σ_t only selects which of its per-position outputs are read, not what they are. Under Assumption 1 the second factor is $\prod_{\ell \in \sigma_t} p_\theta(\boldsymbol{x}^{\ell} \mid \boldsymbol{x}^{\sigma_{<t}})$. Each factor in Equation (8) is a normalized conditional and the process terminates after T steps, so $p_\theta(\cdot, \cdot)$ is a probability distribution and p_θ^π is its $\boldsymbol{x}$-marginal. □

The policy appears only through the first factor. Two consequences are used in the main text. If π ignores the revealed tokens, $\prod_t \pi(\sigma_t \mid \boldsymbol{x}^{\sigma_{<t}})$ is a constant $p(\sigma)$ and $p_\theta^\pi(\boldsymbol{x}) = \mathbb{E}_{\sigma \sim p}\, p_\theta(\boldsymbol{x} \mid \sigma)$, a mixture over orders. If π reads them, no such factorization exists and the sum in Equation (8) runs over the joint, since the denoiser conditions on a different context under each order.

B.2 Ancestral sampling realizes the uniform policy

Proposition 2 (Ancestral sampling is uniform). *Run the reverse process of Section 2.1 ancestrally and record the order σ in which positions are revealed. Then, in the sequential case,*

$$\pi\big(\sigma_t \mid \boldsymbol{x}^{\sigma_{<t}}\big) \;=\; \pi^{\mathrm{unif}}\big(\sigma_t \mid \boldsymbol{x}^{\sigma_{<t}}\big) \;=\; \frac{1}{L-t+1}, \qquad \Pr[\sigma] = \frac{1}{L!} \quad \textit{for every } \sigma \in \mathcal{S}_L,$$

the policy being independent of the revealed tokens. For a step that reveals j positions out of m still masked, the revealed subset is uniform over the $\binom{m}{j}$ subsets.

Proof. Let $U_1, \dots, U_L \sim \mathrm{Unif}[0,1]$ be independent and set $\boldsymbol{z}_t^\ell = \boldsymbol{x}^\ell$ if $U_\ell \le \alpha_t$ and $\mathtt{m}$ otherwise. This realizes the forward process jointly for all t: each position is masked independently with probability $1-\alpha_t$, and, α being decreasing, never unmasked again. It therefore also realizes the time reversal. Run backwards, position ℓ is revealed when α_t reaches U_ℓ, so the unmasking order is the rank order of $(U_1, \dots, U_L)$: uniform on $\mathcal{S}_L$ and independent of $\boldsymbol{x}$, whence $\Pr[\sigma] = 1/L!$ and $\Pr[\sigma_t \mid \sigma_{<t}] = 1/(L-t+1)$. The learned reverse process replaces the unknown $\boldsymbol{x}$ by the denoiser's prediction but keeps the position mechanism: every masked position is revealed independently with the same probability $\gamma_{s|t} = (\alpha_s - \alpha_t)/(1-\alpha_t)$, so conditional on j of the m masked positions being revealed the subset is uniform over the $\binom{m}{j}$ choices, and the argument applies unchanged. □

B.3 The ELBO is the uniform-order NLL

The identity in Equation (12) is known: it is the MDM–AO-ARM equivalence of Ou et al. (2024), Zheng et al. (2024) and Kim et al. (2025a). We give a self-contained counting proof because the bound Equation (13), which reads the identity relative to the family in Equation (2), is what Section 3 builds on and is not stated in those papers.

We use the standard continuous-time setup of Section 2.1. The noise schedule $\alpha : [0,1] \to [0,1]$ is continuously differentiable and strictly decreasing with $\alpha_0 = 1$ and $\alpha_1 = 0$; the forward marginal $q_{t|0}(\boldsymbol{z}_t \mid \boldsymbol{x})$ masks each position independently with probability $1-\alpha_t$, so its revealed set $S(\boldsymbol{z}_t)$ satisfies

$$\Pr\big[S(\boldsymbol{z}_t) = S\big] \;=\; \alpha_t^{|S|}\,(1-\alpha_t)^{L-|S|}. \tag{9}$$

In Equation (1) we take the weight in its non-negative form, $w_t = -\alpha_t'/(1-\alpha_t) \ge 0$, so that each summand is a cross-entropy and $\mathcal{L}_{\mathrm{ELBO}} \ge 0$. The denoiser is time-free (Sahoo et al., 2024; Zheng et al., 2024): $p_\theta(\cdot \mid \boldsymbol{z}_t)$ depends on $\boldsymbol{z}_t$ only through $\boldsymbol{x}^{S(\boldsymbol{z}_t)}$.

Proposition 3 (The ELBO is a weighted sum over every infilling problem). *Under the assumptions above,*

$$\mathcal{L}_{\mathit{ELBO}}(\theta;\boldsymbol{x}) \;=\; \sum_{S \subsetneq [L]} \;\sum_{\ell \in [L]\setminus S} c_{|S|} \cdot \Big(-\log p_\theta\big(\boldsymbol{x}^\ell \mid \boldsymbol{x}^S\big)\Big), \qquad c_n \;=\; \binom{L}{n}^{-1} (L-n)^{-1}, \tag{10}$$

and $c_n = n!\,(L-n-1)!\,/\,L!$. The weight depends on S only through $|S|$, and the weights sum to L.

Proof. The denoiser is time-free, so the inner sum of Equation (1) depends on $\boldsymbol{z}_t$ only through $S(\boldsymbol{z}_t)$. Averaging over the mask pattern with Equation (9) and writing the expectation over t as an integral,

$$\mathcal{L}_{\mathrm{ELBO}}(\theta;\boldsymbol{x}) \;=\; \sum_{S \subsetneq [L]} \;\sum_{\ell \notin S} \left[\int_0^1 w_t\, \alpha_t^{|S|} (1-\alpha_t)^{L-|S|}\, \mathrm{d}t \right] \Big(-\log p_\theta(\boldsymbol{x}^\ell \mid \boldsymbol{x}^S)\Big),$$

the term $S = [L]$ being empty. Substituting $\alpha = \alpha_t$, for which $w_t\,\mathrm{d}t = -\mathrm{d}\alpha/(1-\alpha)$ and the limits swap, the bracket with $n = |S|$ becomes $\int_0^1 \alpha^n (1-\alpha)^{L-n-1}\,\mathrm{d}\alpha = \mathrm{B}(n+1, L-n) = n!\,(L-n-1)!/L! = c_n$. There are $\binom{L}{n}(L-n)$ pairs (S,ℓ) of each size, so the weights sum to $\sum_{n=0}^{L-1} 1 = L$. □

Every pair (S,ℓ) appears, with a strictly positive weight: one network is asked to solve all $L \cdot 2^{L-1}$ infilling problems at once, not the L prefix problems an AR model faces (Ou et al., 2024; Kim et al.,

2025a). This is Proposition 3.1 of Zheng et al. (2024) and Proposition 2.1 of Kim et al. (2025a); we restate it because Proposition 4 is a re-weighting of exactly this sum.

Write $p_\theta(\boldsymbol{x} \mid \sigma)$ for the likelihood along a single order,

$$\log p_\theta(\boldsymbol{x} \mid \sigma) \;=\; \sum_{t=1}^{L} \log p_\theta\big(\boldsymbol{x}^{\sigma_t} \mid \boldsymbol{x}^{\sigma_{<t}}\big). \tag{11}$$

Proposition 4 (The ELBO evaluates the uniform member)**.** *Under the assumptions of Proposition 3, with $p_\theta(\boldsymbol{x} \mid \sigma)$ as in Equation (11),*

$$\mathcal{L}_{\mathit{ELBO}}(\theta; \boldsymbol{x}) \;=\; \mathbb{E}_{\sigma\sim\mathrm{Unif}(\mathcal{S}_L)}\Big[-\log p_\theta(\boldsymbol{x} \mid \sigma)\Big] \;=\; \mathbb{E}_{\sigma\sim\mathrm{Unif}(\mathcal{S}_L)}\left[\sum_{t=1}^{L} -\log p_\theta\big(\boldsymbol{x}^{\sigma_t} \mid \boldsymbol{x}^{\sigma_{<t}}\big)\right]. \tag{12}$$

Consequently, with π^{unif} the policy of Proposition 2,

$$\mathcal{L}_{\mathit{ELBO}}(\theta; \boldsymbol{x}) \;\geq\; -\log \mathbb{E}_{\sigma\sim\mathrm{Unif}(\mathcal{S}_L)}\big[p_\theta(\boldsymbol{x} \mid \sigma)\big] \;=\; -\log p_\theta^{\pi^{\mathrm{unif}}}(\boldsymbol{x}), \tag{13}$$

with equality if and only if $\log p_\theta(\boldsymbol{x} \mid \sigma)$ is constant over $\sigma \in \mathcal{S}_L$.

Proof. For the identity, expand the right-hand side of Equation (12) as

$$\frac{1}{L!} \sum_{\sigma\in\mathcal{S}_L} \sum_{t=1}^{L} -\log p_\theta(\boldsymbol{x}^{\sigma_t} \mid \boldsymbol{x}^{\sigma_{<t}}).$$

The map $(\sigma, t) \mapsto (\sigma_{<t}, \sigma_t)$ hits every pair (S, ℓ) with $\ell \notin S$, and a pair with $|S| = n$ is hit by $n!\,(L-n-1)!$ permutations, those that order S first and place ℓ next. The sum is therefore $\sum_{S\subsetneq[L]} \sum_{\ell\notin S} c_{|S|}\,(-\log p_\theta(\boldsymbol{x}^\ell \mid \boldsymbol{x}^S))$ with $c_n = n!\,(L-n-1)!/L!$, which is Equation (10).

For the bound, take $\pi = \pi^{\mathrm{unif}}$ in Equation (8). By Propositions 1 and 2, $\prod_t \pi^{\mathrm{unif}}(\sigma_t \mid \boldsymbol{x}^{\sigma_{<t}}) = 1/L!$, so $p_\theta(\boldsymbol{x}, \sigma) = p_\theta(\boldsymbol{x} \mid \sigma)/L!$ and $p_\theta^{\pi^{\mathrm{unif}}}(\boldsymbol{x}) = \mathbb{E}_{\sigma\sim\mathrm{Unif}(\mathcal{S}_L)}[p_\theta(\boldsymbol{x} \mid \sigma)]$. Jensen's inequality for the concave logarithm gives $\log p_\theta^{\pi^{\mathrm{unif}}}(\boldsymbol{x}) \geq \mathbb{E}_\sigma[\log p_\theta(\boldsymbol{x} \mid \sigma)] = -\mathcal{L}_{\mathrm{ELBO}}(\theta; \boldsymbol{x})$, with equality if and only if $p_\theta(\boldsymbol{x} \mid \sigma)$ is constant over $\mathcal{S}_L$. □

The identity says what Equation (1) *is*: the negative log-likelihood of $\boldsymbol{x}$ under a uniformly random order, so one network is every σ-learner of Kim et al. (2025a) at once. The bound says what it is *relative to* the family in Equation (2): a Jensen bound on the π^{unif} member, whose slack is the spread of $\log p_\theta(\boldsymbol{x} \mid \sigma)$ over orders. For π^{unif} the policy factor is the constant $1/L!$ and drops out of Equation (13); a policy that reads the revealed tokens leaves a policy term in any such bound, which is what planner-aware bounds must carry, while for π^F the sum has a single term and the inequality is an identity (Theorem 1).

B.4 Proof of the exact-likelihood theorem

We prove Theorem 1 for $k = 1$, where $\sigma \in \mathcal{S}_L$ is a permutation and F returns one position per step; $k > 1$ follows at the end. The proof uses only what the definitions already give: the denoiser reads a partially revealed sequence and nothing else (Section 2.1), and the rule F is a deterministic function of that same sequence (Definition 2).

Theorem 1 (restated). *Let (p_θ, F) be a* DUEL *sampler and let $\boldsymbol{x} \in \mathcal{V}^L$ be any sequence. We can compute the* exact *likelihood of $\boldsymbol{x}$ under $p_\theta^{\pi^F}$ as:*

$$\log p_\theta^{\pi^F}(\boldsymbol{x}) \;=\; \sum_{t=1}^{T} \sum_{\ell\in\sigma_t^F} \log p_\theta\big(x^\ell \mid \boldsymbol{x}^{\sigma_{<t}^F}\big),$$

for the order $\sigma^F(\boldsymbol{x})$ that F produces on $\boldsymbol{x}$ and its number of steps T. Algorithm 2 implements the procedure with one forward pass per step, $T \leq L$ passes in total.

Define $\sigma^F(\boldsymbol{x})$ by the recursion

$$\sigma^F_{<1} = \emptyset, \qquad \sigma^F_t = F\big(\boldsymbol{x}^{\sigma^F_{<t}}\big), \qquad \sigma^F_{<t+1} = \sigma^F_{<t} \cup \sigma^F_t, \qquad t = 1, \dots, L. \tag{14}$$

Proof of Theorem 1. Step 1: $\sigma^F(\boldsymbol{x})$ exists and is unique. Each right-hand side of Equation (14) depends only on quantities fixed at step t, and F returns a masked position, so the recursion runs for exactly L steps and produces one permutation. It depends on $\boldsymbol{x}$, and is not known before F has been run.

Step 2: every other order has zero mass. Let $\sigma \neq \sigma^F(\boldsymbol{x})$ first differ from $\sigma^F(\boldsymbol{x})$ at step t. The two share the revealed prefix, so $\boldsymbol{x}^{\sigma_{<t}} = \boldsymbol{x}^{\sigma^F_{<t}}$ and F returns the same position for both: $F(\boldsymbol{x}^{\sigma_{<t}}) = \sigma^F_t \neq \sigma_t$. By Equation (5), $\pi^F(\sigma_t \mid \boldsymbol{x}^{\sigma_{<t}}) = 0$, and the product in Equation (2) vanishes. Only $\sigma = \sigma^F(\boldsymbol{x})$ survives the sum, and each of its policy factors equals 1:

$$p_\theta^{\pi^F}(\boldsymbol{x}) \;=\; \sum_{\sigma \in \mathcal{S}_L} p_\theta(\boldsymbol{x}, \sigma) \;=\; \prod_{t=1}^{L} \underbrace{\pi^F\big(\sigma^F_t \mid \boldsymbol{x}^{\sigma^F_{<t}}\big)}_{=1} \cdot p_\theta\big(\boldsymbol{x}^{\sigma^F_t} \mid \boldsymbol{x}^{\sigma^F_{<t}}\big).$$

Taking logarithms gives Equation (6). □

Step 2 needs only that two orders agreeing on the revealed prefix receive the same output from F; Appendix C uses this to admit stateful rules.

Parallel unmasking, $k > 1$. For $k > 1$ the order is an ordered partition $(\sigma_1, \dots, \sigma_T)$ of $[L]$ (Definition 1), with σ_t the set revealed at step t. Under a fixed-k rule $|\sigma_t| = \min(k, |\mathcal{M}|)$ (Appendix C), so the last step of a sequence or block reveals the remainder, and $T = \lceil L/k \rceil$. The proof goes through with $\mathcal{S}_L$ replaced by the set of ordered partitions and $\boldsymbol{x}^{\sigma_t}$ read as the tokens at the whole set σ_t; the token factor then splits over $\ell \in \sigma_t$ by Assumption 1, which holds by construction of Algorithm 1 (Appendix B.1). Theorem 1 at $k > 1$ is therefore exact for the sampler's own distribution. It is not a claim about the data: intra-step dependencies are not represented, so raising k moves $p_\theta^{\pi^F}$ to a different, empirically worse, member of the family, as the k-ladder in Appendix E.1 shows.

C UNMASKING RULES

An unmasking rule decides which masked positions to reveal next. The denoiser already tells the sampler where it is confident, and revealing those positions first gives every later prediction a more reliable context; this is why practical rules read the denoiser's output rather than the clock. Revealing positions uniformly at random discards that information. It is the policy training averages over (Proposition 4), not a policy anyone would choose to generate with.

The rules we use. Table 4 defines the five rules of Section 3 formally. Fix a step and write $\mathcal{M} = \mathcal{M}(\boldsymbol{x}^{\sigma<t}) \subseteq [L]$ for the masked positions, $\boldsymbol{P} = p_\theta(\cdot \mid \boldsymbol{x}^{\sigma<t})$ for the denoiser's output, and $P_\ell^{(1)} \geq P_\ell^{(2)}$ for the largest and second-largest entries of P_ℓ. Three conventions make every rule a total, deterministic function, which is what Theorem 1 needs. A fixed-k rule returns $k' = \min(k, |\mathcal{M}|)$ positions, since fewer than k may remain at the last step of a sequence or block. $\mathrm{top}_j(s)$ takes the j positions with the largest scores and breaks ties by lowest index, so that F is a function rather than a correspondence. Threshold rules select by a test that no position need pass, so they fall back to the single most confident position when the test selects nothing:

$$\langle \mathcal{A} \rangle \triangleq \begin{cases} \mathcal{A}, & \mathcal{A} \neq \emptyset, \\ \mathrm{top}_1\big(P^{(1)}\big), & \mathcal{A} = \emptyset. \end{cases} \tag{15}$$

Stateful rules. KLASS compares the current prediction with the previous step's, so it is a function of the trajectory rather than of $\boldsymbol{x}^{\sigma<t}$ alone. Let the sampler carry state $h_t = G(\boldsymbol{x}^{\sigma<t}, h_{t-1})$ and let F act on $(\boldsymbol{x}^{\sigma<t}, h_{t-1})$. Since h_{t-1} is a function of the trajectory up to step $t-1$, two orders that agree on every step before t receive the same output from F, which is all Step 2 of the proof of Theorem 1 uses. At $t = 1$ we set $\hat{P}_\ell = P_\ell$, so KLASS coincides with confidence threshold on the first step.

Other deterministic rules. The table is a sample, not a census. Entropy-based selection (Ben-Hamu et al., 2025), learned position policies (Jazbec et al., 2025), schedules for how many positions to reveal at each step (Luxembourg et al., 2025; Hayakawa et al., 2025), and any of these composed with block decoding (Arriola et al., 2025a) are deterministic functions of the revealed tokens, so Theorem 1 applies to each without change. What falls outside are rules that inject randomness into position selection, such as sampling positions in proportion to confidence, and remasking samplers (Wang et al., 2025a; Huang et al., 2025) that revisit revealed positions.

Nobody samples uniformly at random. The uniform policy π^{unif} is what the training objective averages over, and the ELBO is invariant to the policy used at inference, so standard evaluation never had to record which policy a model was run with. Generation is another matter. Every large-scale MDM report we are aware of produces its headline results with a deterministic rule: LLaDA with its low-confidence remasking rule, greedy confidence in our terms (Nie et al., 2025), Dream (Ye et al., 2025), LLaDA-2.0 (Bie et al., 2025), SDAR (Cheng et al., 2025), and Nemotron-Diffusion (Fu et al., 2026) with confidence-, entropy- or threshold-based rules. Where uniformly random unmasking

Table 4: The unmasking rules, as set-valued functions of the revealed tokens. $\mathcal{M} = \mathcal{M}(\boldsymbol{x}^{\sigma<t})$; $k' = \min(k, |\mathcal{M}|)$; top_j breaks ties by lowest index; $\langle\cdot\rangle$ is the argmax fallback of Equation (15). μ is a confidence threshold, ν a stability threshold, and $\hat{P}_\ell$ the previous step's prediction at ℓ. Every row is a deterministic function of quantities computable from $\boldsymbol{x}^{\sigma<t}$, so Theorem 1 applies to all five, and $|F| \geq 1$ always. The intuitions are in Section 3.

Rule F	$F(\boldsymbol{x}^{\sigma<t})$	$\lvert F\rvert$
Left-to-right	{the k' smallest $\ell \in \mathcal{M}$}	k'
Greedy confidence (Nie et al., 2025)	$\mathrm{top}_{k'}\big(P_\ell^{(1)}\big)$	k'
Probability margin (Kim et al., 2025a)	$\mathrm{top}_{k'}\big(P_\ell^{(1)} - P_\ell^{(2)}\big)$	k'
Confidence threshold (Wu et al., 2025)	$\big\langle \{\ell \in \mathcal{M} : P_\ell^{(1)} \geq \mu\} \big\rangle$	adaptive
KLASS (Kim et al., 2025b)	$\big\langle \{\ell \in \mathcal{M} : P_\ell^{(1)} \geq \mu,\ D_{\mathrm{KL}}(\hat{P}_\ell \,\|\, P_\ell) \leq \nu\} \big\rangle$	adaptive

appears at all, it is as an ablation baseline that degrades quality (Nie et al., 2025; Kim et al., 2025a). The policy the ELBO scores is therefore one that no deployed sampler draws from, which is the mismatch Section 3 removes.

D EXPERIMENTS

D.1 PERPLEXITY

Details for Section 4.1. Training and architecture follow Sahoo et al. (2024) and Arriola et al. (2025a).

Data. In-domain: OpenWebText (OWT; Gokaslan et al., 2019) with the GPT-2 tokenizer (Radford et al., 2019), and One Billion Words (LM1B; Chelba et al., 2013) with the BERT tokenizer (Devlin et al., 2019). Zero-shot: models trained on OWT are evaluated on Penn Treebank (Marcus et al., 1993), Wikitext (Merity et al., 2016), LM1B, Lambada (Paperno et al., 2016), AG News (Zhang et al., 2015), and the PubMed and ArXiv subsets of Scientific Papers (Cohan et al., 2018). All sequences are concatenated and wrapped to the model's context length following Arriola et al. (2025a), and every evaluation uses the full validation/test split. The 8B models are evaluated on Wikitext, Lambada and AG News.

Models. The 110M models are 12-layer DiTs, hidden dimension 768, 12 heads, trained for 1M steps at batch size 512, with $L = 1024$ on OWT and $L = 128$ on LM1B; the AR baseline, SEDD, MDLM and the three BD3-LMs are therefore directly comparable. At scale we evaluate LLaDA-8B-Base (Nie et al., 2025) at $L = 2048$, with Llama3-8B-Base (Grattafiori et al., 2024) as a reference point only: the two differ in training data and compute budget, so no gap is computed for that pair.

Likelihood estimators. Three estimators are used throughout, one per column or row label of every table.

- **AR** — exact under δ_{σ_0}. The chain rule under the left-to-right order, $\log p(\boldsymbol{x}) = \sum_{\ell=1}^{L} \log p_\theta(\boldsymbol{x}^\ell \mid \boldsymbol{x}^{\sigma<\ell})$ with σ the identity, computed in one causal forward pass.
- **ELBO** — a bound under π^{unif} (MDMs). A Monte Carlo estimate of the ELBO, averaged over uniformly sampled corruption levels and masked sets. For SEDD and MDLM each masked position is weighted by L divided by the number of positions masked in that draw, and one Monte Carlo sample is one full forward pass at length L. For BD3-LM the objective factorizes over blocks with all preceding blocks revealed (Arriola et al., 2025a, Eq. 8); the implementation concatenates the corrupted and clean sequences into a single input of length $2L$ under a block-diffusion attention mask, so *one* forward pass at length $2L$ yields all L/L' blocks at once — L/L' stratified draws per sequence (256, 128, 64 at $L' = 4, 8, 16$) for the price of one forward, against a single draw for MDLM and SEDD. We use $\text{MC} = 1$ Monte Carlo sample with a low-discrepancy sampler (Kingma et al., 2021) at 110M and $\text{MC} = 128$ for LLaDA-8B. The two settings are consistent rather than inconsistent: in the 110M codebase the batch axis carries distinct sequences, so M samples cost M full validation passes, whereas in the LLaDA codebase the batch axis *is* the Monte Carlo axis. Four repeats of the 110M estimate give a relative standard deviation of 2.2–2.8% for all four MDM arms; the extra stratification BD3-LM gets for free does *not* show up as lower corpus-level variance, because at this sample size the aggregate is dominated by across-sequence variation.
- **DUEL** — exact under π^F. The exact likelihood under the deterministic policy, obtained by summing $\log p_\theta(\boldsymbol{x}^{\sigma_t} \mid \boldsymbol{x}^{\sigma<t})$ along the single order the rule F induces on $\boldsymbol{x}$.

Table 5: Zero-shot perplexity (↓) for 110M models trained on OWT. AR is exact under δ_{σ_0}, the ELBO a bound under π^{unif}, DUEL exact under π^F. **Bold**: better of ELBO/DUEL per model–corpus pair.

Model	Estimator	PTB	Wikitext	LM1B	Lambada	AG News	PubMed	ArXiv
AR	exact	81.07	25.32	51.14	52.13	52.11	48.59	41.22
SEDD	ELBO	≤96.33	≤35.98	≤68.14	≤48.93	≤67.82	≤45.93	≤40.52
	DUEL	**91.56**	**31.63**	**64.39**	**46.01**	**63.78**	**39.31**	**35.48**
MDLM	ELBO	≤90.96	≤33.22	≤64.94	≤48.29	≤62.78	≤43.64	≤38.35
	DUEL	**87.57**	**30.97**	**60.94**	48.42	**59.81**	**41.96**	**37.57**
BD3-LM	ELBO	≤96.81	≤31.31	≤60.88	≤50.03	≤61.67	≤42.52	≤39.20
L'=4	DUEL	**83.94**	**29.43**	**57.98**	**48.56**	**56.73**	**41.41**	**38.11**

Table 6: Per-budget IsoFLOP vertices and fitted frontier exponents on 800 held-out SlimPajama sequences. $(N_{\text{opt}}, L_{\text{opt}})$ is the vertex of the quadratic in $\log N$; a is the OLS slope of $\ln L_{\text{opt}}$ on $\ln C$, $\pm$ its standard error. DUEL is exact under left-to-right unmasking; the ELBO a bound under π^{unif}. Last row: whether the *discrete* best checkpoint agrees under both. At 30×10^{18} the ELBO picks 231M by 0.0002 nats and DUEL 336M by 0.0051 nats; refitting on that checkpoint moves a by 0.03σ and the gap by $0.05\times$.

Estimator		**Compute budget C ($\times 10^{18}$ FLOPs)**					
		6	**10**	**30**	**60**	**100**	**Exponent a**
AR	N_{opt} (M)	156	221	439	622	843	0.0645 ± 0.0009
	L_{opt}	2.918	2.818	2.620	2.508	2.436	
MDM ELBO	N_{opt} (M)	83	110	229	362	478	0.0626 ± 0.0010
	L_{opt}	3.449	3.339	3.110	2.977	2.897	
MDM DUEL (left-to-right)	N_{opt} (M)	79	115	227	386	496	0.0661 ± 0.0011
	L_{opt}	3.398	3.279	3.044	2.908	2.825	
ELBO / DUEL argmin agrees		✓	✓	✗	✓	✓	

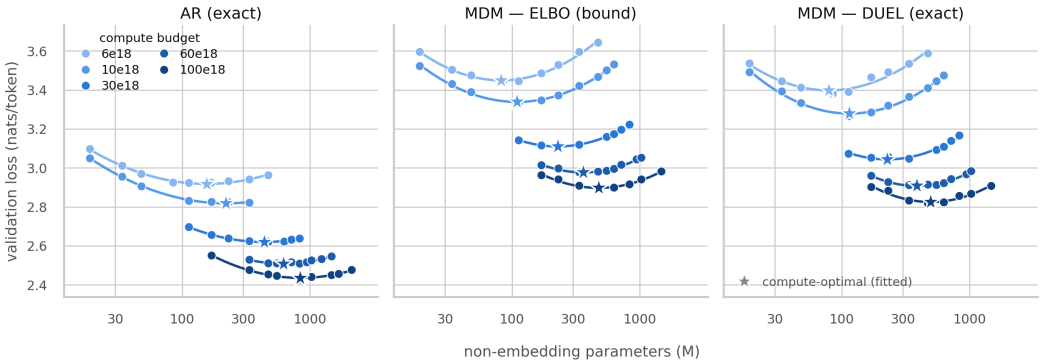

Figure 4: IsoFLOP parabolas behind Tables 2 and 6: loss against non-embedding parameters N at fixed budget C ($C = 6ND$), stars at the fitted vertices. Center and right are the *same* MDM checkpoints on the *same* 800 sequences, scored by the ELBO (π^{unif}, bound) and by DUEL at $k = 1$ left-to-right (exact); the right panel sits $\approx$0.07 nats lower with a larger frontier exponent, which is the whole of the $\times 16 \to \times 9$ revision.

Each estimator returns $\text{NLL} = -\log p(\boldsymbol{x})$, reported as $\text{PPL} = \exp(\text{NLL}/L)$.

DUEL protocol. All 110M runs use block-greedy unmasking with $k = 1$ token revealed per step. Each BD3-LM is evaluated at its native training block size, $L' \in \{4, 8, 16\}$. SEDD and MDLM are trained globally and have no native block size, so the blocking is purely an evaluation choice; we set $L' = 4$ to match the smallest BD3-LM for a like-for-like comparison. LLaDA-8B-Base uses block-greedy unmasking with $k = 1$ and $L' = 32$.

Where the metric is defined. The gap closed (Section 4.1) requires $\Delta_{\text{ELBO}} > 0$. On Lambada, PubMed and ArXiv the MDM already outperforms the AR baseline under the ELBO, so $\Delta_{\text{ELBO}} \leq 0$ and the ratio is meaningless; those three corpora are excluded from Table 1c but appear in Table 5, where DUEL still delivers reductions of comparable magnitude. DUEL is lower than the ELBO on every model–corpus pair in Table 5 except MDLM on Lambada, where the two are within the ELBO's sampling noise: DUEL $= 48.42$ against a single ELBO draw of $\leq$48.29 whose mean over 12 seeds is 48.45 ± 0.20. Nothing forces DUEL $<$ ELBO in any case: the two score $p_\theta^{\pi^F}$ and $p_\theta^{\pi^{\text{unif}}}$, which are different distributions.

D.2 SCALING LAWS

Details for Section 4.2.

Checkpoints and budgets. We use Nie et al.'s released AR and MDM IsoFLOP grids without retraining or modification: non-embedding parameter counts N from 19M to 1028M, at the five compute budgets $C \in \{6, 10, 30, 60, 100\} \times 10^{18}$ FLOPs, under their convention $C = 6ND$ relating budget, size and tokens seen D. Each (budget, size) pair is one released checkpoint, giving the parabolas of Figure 4; every budget carries enough sizes to bracket its minimum on both sides, and no fitted vertex fell at the edge of its bracket.

Validation data. All three losses are computed by us, on one fixed set of 800 held-out SlimPajama sequences—the same sequences, the same tokenization, for every checkpoint and every metric. Nothing is read off the published tables, so a level offset in any one metric cannot be mistaken for a difference between metrics.

The three losses. Per checkpoint:

- **AR**: the exact autoregressive likelihood, one forward pass per sequence. Policy δ_{σ_0}, exact.
- **MDM ELBO**: the standard masked-diffusion bound with $\mathrm{MC} = 128$ Monte Carlo samples, the Nie et al. (2024) default. Policy π^{unif}, bound.
- **MDM DUEL**: the exact likelihood at $k = 1$ under left-to-right unmasking, δ_{σ_0} with $\sigma_0 = (1, \ldots, L)$. One forward pass per unmasking step. Policy δ_{σ_0}, exact.

The ELBO and DUEL evaluate the *same* checkpoints on the *same* sequences, so their per-sequence difference is paired: its standard error is far smaller than that of either loss taken separately. Left-to-right is chosen over the confidence rule used in the other experiments because it is the order closest to AR generation, which is what makes the two sides of the comparison commensurable; it is a fixed order, hence δ_{σ_0}, not π^F.

Fitting. Nie et al.'s procedure, with only the y-axis changed. Their paper does not say whether L_{opt} is the fitted vertex or the best trained model; we read the former off their published figures, where the plotted L_{opt} lies below every checkpoint at that budget. *(i)* For each metric and budget, fit an unweighted quadratic of loss against $\log N$ over the sizes at that budget and take its vertex as the compute-optimal pair $(N_{\mathrm{opt}}, L_{\mathrm{opt}})$; the vertex fell strictly inside the sampled size range for all fifteen fits, so no bracket had to be extended. *(ii)* Fit $L_{\mathrm{opt}} \propto C^{-a}$ across the five budgets by OLS in log-log space. The one budget at which the ELBO and DUEL select different checkpoints, 30×10^{18}, is also the one with the least leverage on the frontier slope: its $\ln C$ sits almost exactly on the centroid of the five budgets (hat value 0.20, against 0.53–0.57 at the endpoints), so moving that point shifts the intercept and not the exponent. This yields, for each metric m, a straight-line frontier

$$\ln L_m(C) = \ln A_m - a_m \ln C, \qquad m \in \{\mathrm{AR},\ \mathrm{ELBO},\ \mathrm{DUEL}\}. \tag{16}$$

Fitted exponents and per-budget vertices are in Table 6.

The equal-loss compute gap. The gap at budget C is the *horizontal* distance between the MDM and AR frontiers: the budget $x_m(C)$ at which the MDM reaches the AR model's compute-optimal loss, divided by C. Writing that out from Equation (16),

$$\begin{aligned} L_m\big(x_m(C)\big) &= L_{\mathrm{AR}}(C), \qquad r_m(C) \;=\; x_m(C)/C, \\ r_m(C) \;&=\; \exp\!\left(\frac{\ln A_m - \ln A_{\mathrm{AR}}}{a_m} \;+\; \left(\frac{a_{\mathrm{AR}}}{a_m} - 1 \right) \ln C \right). \end{aligned} \tag{17}$$

Each frontier is inverted with *its own* exponent a_m. The gap is therefore not constant in C: by Equation (17) it grows with C when $a_m < a_{\mathrm{AR}}$ and shrinks when $a_m > a_{\mathrm{AR}}$. Since $a_{\mathrm{ELBO}} < a_{\mathrm{AR}} < a_{\mathrm{DUEL}}$, the ELBO gap widens with scale and the DUEL gap narrows—which is why the correction in Table 2 grows from -31% to -41%.

The naive route is wrong. It is tempting to convert the loss difference at a single budget into a compute ratio by walking it back along the *AR* frontier alone,

$$\tilde{r}_m(C) = \exp\left(\frac{\ln L_m(C) - \ln L_{\mathrm{AR}}(C)}{a_{\mathrm{AR}}}\right). \tag{18}$$

Equation (18) equals Equation (17) only when $a_m = a_{\mathrm{AR}}$. Here it does not: with the fitted exponents of Table 6 it understates the ELBO gap by $\approx 8\%$ and overstates the DUEL gap by $\approx 6\%$. It also divides a level difference by a single constant, so it makes $\tilde{r}_m$ nearly flat in C and hides the growth of the gap cut. All numbers reported in Section 4.2 use Equation (17).

D.3 Multiple-choice benchmarks

Model and scorers. LLaDA-8B-Base, no classifier-free guidance ($\mathrm{cfg} = 0$), `fp32` log-probabilities on both arms, one RTX A6000 per arm. The ELBO uses $\mathrm{MC} = 128$ stratified samples in four batch-32 rounds, LLaDA's published protocol. DUEL uses $k = 1$ with block size 32, under the greedy rule (π^F) and under left-to-right unmasking (δ_{σ_0}). Accuracies are on the full split; the timings of Table 14 use a fixed per-task subsample of 200–500 documents, with both arms run back-to-back in one job on the same GPU. Because our runs set $\mathrm{cfg} = 0$, absolute accuracies are not directly comparable to the numbers in the LLaDA paper, which are tuned over a per-task guidance scale; every comparison here is between estimators on the same model and the same requests.

Tasks. The six tasks are `piqa`, `hellaswag`, `mutual_mc`, `lsat_lr_mc`, `agieval_logiqa_en` and `agieval_lsat_rc`. At $k = 1$ the number of forwards DUEL spends on a choice equals the number of tokens in that choice, so the NFE column of Table 14 is also the mean answer length: 18.3–29.4 tokens. Tasks whose answers are much shorter are uninformative for this comparison rather than unfavorable to it: with few masked positions there is almost no ordering for the ELBO to get wrong, and the two estimators nearly coincide. In our inventory of LLaDA multiple-choice tasks the mean answer length is 1.00 tokens on `commonsense_qa` and on 5-shot `mmlu` (a single scored token, a structural null for any ordering effect), 3.36 on `openbookqa`, 4.28 on `mathqa`, 6.00 on `arc_challenge`, 7.61 on `copa`, 8.87 on `storycloze` and 10.65 on `swag`. Those tasks are where DUEL's *cost* advantage is largest — fewer answer tokens means fewer forwards — so excluding them is conservative on cost and neutral-to-conservative on accuracy.

Metric convention. LLaDA does not state which of `acc` and `acc_norm` it reports; the convention has to be inferred from its baselines, which were run by its authors under the same harness. We follow the inferred convention: `acc_norm` on `hellaswag`, `acc` on the other five. Length normalization matters most on `hellaswag` (where the two differ by $\sim$19 points on the LLaMA-2 reference numbers) and the choice is applied identically to all three scorers, so it cannot favor one of them.

Temperature scaling and cross-fitting. For document d with choice set $\mathcal{C}_d$ and correct choice y_d, a scorer produces raw log-likelihoods $s_d(c)$, $c \in \mathcal{C}_d$. These are *sums* of token log-probabilities, so a softmax over them is wildly overconfident, and the three scorers differ in scale as well as in ranking; untempered, any probabilistic metric would mostly measure that scale. We therefore fit a single temperature $T > 0$ per task and per scorer and set

$$p_d(c) \;=\; \frac{\exp\big(s_d(c)/T\big)}{\sum_{c' \in \mathcal{C}_d} \exp\big(s_d(c')/T\big)}.$$

T is fit by minimizing the negative log-likelihood of the correct choice, using 2-fold cross-fitting: the documents of a task are split in half, a temperature is fit on each half, and every document is scored with the temperature fit on the half that *excludes* it. No document contributes to the temperature used to score it, and accuracy is unaffected — T is a monotone rescaling and does not move any argmax.

Metrics. With p_d as above, and N documents per task,

$$\text{Brier} = \frac{1}{N}\sum_d \sum_{c \in \mathcal{C}_d} \big(p_d(c) - \mathbf{1}[c = y_d]\big)^2, \qquad \text{NLL} = -\frac{1}{N}\sum_d \log p_d(y_d), \tag{19}$$

Table 7: Per-task calibration of the predicted distribution over answer choices, after per-task, per-scorer temperature scaling with 2-fold cross-fitting; l2r = left-to-right. The ELBO is a bound under π^{unif}; both DUEL columns are exact. Best per task in bold.

Task	Brier ↓			NLL ↓			AUROC ↑		
	ELBO	DUEL (greedy)	DUEL (l2r)	ELBO	DUEL (greedy)	DUEL (l2r)	ELBO	DUEL (greedy)	DUEL (l2r)
MuTual	0.638	0.628	**0.627**	1.178	1.160	**1.158**	0.595	0.603	**0.612**
LogiQA	0.737	0.718	**0.698**	1.362	1.331	**1.306**	0.569	0.584	**0.636**
LSAT-RC	0.591	0.566	**0.494**	1.210	1.201	**1.053**	0.714	0.719	**0.752**
PIQA	0.420	**0.409**	0.411	0.631	**0.620**	0.623	**0.579**	0.575	0.561
LSAT-LR	0.751	**0.719**	0.724	1.492	**1.456**	1.485	0.503	0.662	**0.665**
HellaSwag	0.626	**0.610**	0.611	1.149	**1.122**	1.124	0.620	**0.639**	**0.639**
Average	0.627	0.608	**0.594**	1.170	1.148	**1.125**	0.597	0.630	**0.644**

both lower-better, both strictly proper, and both bin-free. AUROC is the area under the ROC curve of the top-choice confidence $\max_c p_d(c)$ used as a detector of the scorer's own correctness $\mathbf{1}[\arg\max_c p_d(c) = y_d]$; it measures discrimination — is the scorer more confident on the questions it gets right — and is invariant to T. **We do not report expected calibration error.** ECE is not a proper scoring rule, and it depends on a binning that is unreliable at the few hundred documents per task available here; a scorer can improve its ECE by becoming a worse predictor.

D.4 Sampler selection

Details and full results for Section 4.4; Appendix E.3 checks the same ranking against GSM8K accuracy. The subsection runs in the order of the argument: the setup, why the ELBO cannot rank samplers, the judge and its two arms, the pairwise result, the exhibits, and the prompts.

Generation configurations. 110*M:* BD3-LM (Arriola et al., 2025a) with block size $L' = 16$ trained on OpenWebText (OWT; Gokaslan et al., 2019), GPT-2 tokenizer, $L = 1024$, nucleus $p = 0.9$, unconditional. 8*B:* LLaDA-8B-Base (Nie et al., 2025), nucleus $p = 0.98$, temperature $T = 1.0$, $L = 256$ generated tokens conditioned on a 32-token OWT prefix. We draw 1000 sequences per sampler configuration at both scales. DUEL perplexity is computed on held-out OWT text — arbitrary validation sequences at 110M, and each prefix's exact 256-token continuation at 8B (981 full-length prefixes) — never on generated samples. gen-ppl and token entropy are computed on the generated text under GPT-2 Large (Radford et al., 2019); MAUVE (Pillutla et al., 2021) compares 1000 generated samples to 1000 OWT reference sequences in the same embedding space.

NFE **calibration.** For left-to-right, greedy confidence and probability margin the budget is exact: fixing k tokens per forward pass gives $\lceil L/k \rceil$ evaluations, so $k \in \{1, 2, 4, 8\}$ gives NFE $\in \{1024, 512, 256, 128\}$ at 110M and $\{256, 128, 64, 32\}$ at 8B. Confidence threshold instead unmasks every position whose top-1 probability exceeds μ (or the single most confident position if none do), so its per-sequence NFE is data-dependent and must be calibrated, separately for evaluation and for generation:

- 110M, DUEL perplexity: $\mu \in \{0.05, 0.07, 0.15, 0.99\}$ gives mean NFE $\{140, 242, 514, 1003\}$.
- 110M, generation (gen-ppl, entropy, MAUVE): $\mu \in \{0.045, 0.08, 0.16, 1.0\}$ gives mean NFE $\{131, 278, 518, 921\}$.
- 8B, generation: $\mu \in \{0.028, 0.041, 0.075, 0.99\}$ targets NFE $\in \{32, 64, 128, 256\}$, but the same μ costs only $\{8, 15, 53, 253\}$ evaluations on held-out text — up to a $4\times$ evaluation/generation divergence.

This divergence is why confidence threshold is excluded from Table 10: its DUEL perplexity and its generations are not measured at the same budget, so the comparison is not budget-matched. It is retained in Tables 8 and 9, where each metric is read within its own arm.

Table 8: **DUEL perplexity (↓) by unmasking rule and** NFE, BD3-LM $L' = 16$ on 1,000 held-out OWT sequences (NFE $= L/k$). DUEL rows are exact under π^F; the ELBO, a bound under π^{unif}, is invariant to rule and budget and is shown once, on this subset rather than the full split. **Bold**: best per budget.

Rule F	NFE			
	128	**256**	**512**	**1024**
Left-to-Right	240.27	109.00	44.51	**21.46**
Greedy Conf.	165.48	67.15	34.74	22.04
Prob. Margin	**141.31**	**57.66**	**32.33**	22.35
Conf. Thresh.	227.77	116.95	43.45	22.04
ELBO (reference)		≤ 23.52		

The ELBO cannot rank samplers. The ELBO scores under π^{unif} whatever rule is run, so it returns one number for every sampler; DUEL scores each rule's own policy. Table 8 shows the consequence at 110M: DUEL perplexity separates the four rules at every budget and orders them consistently, probability margin best at low NFE and left-to-right at the full budget, while the ELBO is a single reference row. Only a policy-dependent likelihood can rank samplers; the rest of this appendix checks that DUEL's ranking is right.

Judge protocol. Claude Sonnet 5 with JSON-schema-enforced structured outputs, blinded configuration ids, fixed seed. Two arms. The *absolute* arm rates a single generation 1–10 on fluency, grammaticality, coherence and consistency plus an overall score; the *relative* arm compares two same-budget generations from different samplers and returns a winner plus the single dimension that drove the verdict. The two arms answer different questions. The relative arm is the ground truth for ranking: a blinded A/B choice at matched budget is robust to scale drift and keeps its resolution when both samples are poor, which is the regime at low NFE, where every absolute rating floors at $1/10$ and cannot separate samplers (Table 9). The absolute arm supplies what a binary verdict cannot: per-dimension scores, which give the correlation between DUEL perplexity and each judged dimension (Table 11), and a check that where ratings are off the floor the pairwise winner is also rated higher (Exhibit 2). The 8B study uses the same prompts except that the absolute system prompt says "research language model" instead of "small (110M-parameter) research language model". Factuality is not judged: open-ended web-text continuations offer no reference to check claims against, and factuality reflects model knowledge rather than the unmasking policy.

Absolute judge. The absolute arm rates every configuration on the four dimensions and overall (Table 9). Two things follow. First, DUEL perplexity tracks every judged dimension: the Spearman correlation between DUEL and the judge's mean score is -0.92 to -0.99 per dimension across each scale's 16 configurations (Table 11, left); these full-grid correlations are budget-dominated and establish coverage rather than superiority, since the four dimensions are collinear and gen-ppl reaches -0.94 at 8B on the same grid. Second, the arm floors: from the third budget down every configuration reads 1.00, so absolute ratings cannot separate samplers exactly where the metrics disagree most. Ranking is therefore decided by the pairwise arm.

Pairwise judge. The unit of analysis is one same-index sample pair judged in both A/B presentation orders, scored win $= 1$ / tie $= 0.5$ / loss $= 0$ and order-averaged; $n \approx 500$ units per pair. Intervals are Wilson intervals on the order-averaged units, which is conservative relative to a unit-level bootstrap since the units have sub-Bernoulli variance. Table 10(b) lists every pair with its interval. DUEL agrees with the judge on all 14 decided pairs, against $8/14$ for gen-ppl and $12/14$ for MAUVE (Table 10a); counting the 10 undecided pairs by point direction, DUEL leads $21/24$ against $15/24$ and $16/24$. The verdicts it agrees with are decided by coherence, 79% at 110M and 55% at 8B, not by surface grammar (Table 11, right). gen-ppl is inverted at 110M ($1/7$): left-to-right wins gen-ppl at every parallel budget through entropy collapse (token entropy 3.99–4.55 against ≈ 4.9 for its peers) while the judge ranks it last. It is clean at 8B ($7/7$), where nucleus $p = 0.98$ suppresses the loops. MAUVE fails exactly where sample-based metrics lose dynamic range: both decided 110M NFE $= 128$ pairs read 0.005 for every sampler. The two metrics also differ in what they cost to run —

Table 9: **Judge absolute ratings** (1–10**,** ↑**) per configuration**: overall score with (fluency / grammaticality / coherence / consistency); $n \approx 1000$ per configuration at the top two budgets, $\approx$ 100–128 at the lowest two. From the third budget down every configuration floors at 1.00. Confidence-threshold rows are not budget-matched.

Scale	NFE	**Left-to-Right**	**Greedy Conf.**	**Prob. Margin**	**Conf. Thresh.**
110M (BD3-LM)	1024	**2.37**	2.32	2.32	2.27
		(3.82/4.36/2.70/2.14)	(3.70/4.15/2.55/2.04)	(3.73/4.12/2.54/2.04)	(3.71/4.15/2.67/2.12)
	512	1.00	1.08	1.11	**1.12**
		(1.00/1.00/1.00/1.01)	(1.32/1.31/1.21/1.22)	(1.51/1.50/1.34/1.29)	(1.41/1.36/1.34/1.30)
	256	1.00	1.00	1.00	1.00
	128	1.00	1.00	1.00	1.00
8B (LLaDA)	256	**1.80**	1.70	1.63	1.73
		(2.58/2.86/2.02/1.99)	(2.53/2.83/1.84/1.96)	(2.41/2.73/1.77/1.88)	(2.57/2.90/1.87/1.98)
	128	1.00	1.02	**1.07**	1.01
		(1.02/1.02/1.00/1.02)	(1.07/1.09/1.05/1.08)	(1.20/1.19/1.11/1.16)	(1.03/1.03/1.02/1.03)
	64	1.00	**1.01**	1.00	1.00
	32	1.00	1.00	1.00	1.00

DUEL needs nothing beyond the log-probabilities already produced during generation, gen-ppl needs ~ 1000 samples plus an external LM, and MAUVE needs thousands of samples plus a featurizer.

Table 10: **Agreement with the LLM judge.** (a) Decided pairs each metric agrees on, with all 12 pairs per scale in parentheses. (b) Every pair with its 95% Wilson interval; marks on tie rows are parenthesized and excluded from (a). l2r/gr/mg = left-to-right, greedy confidence, probability margin; D = DUEL, G = gen-ppl, M = MAUVE.

(a) Agreement on decided pairs

Scale	**DUEL**	**gen-ppl**	**MAUVE**
110M (BD3-LM)	**7/7** (10/12)	1/7 (4/12)	5/7 (8/12)
8B (LLaDA)	**7/7** (11/12)	7/7 (11/12)	7/7 (8/12)

(b) All 24 within-budget pairs

NFE	**Pair (judge direction)**	**Win-rate [95% CI]**	**Status**	**D**	**G**	**M**
110**M (BD3-LM** $L' = 16$**, nucleus** $p = 0.9$**)**						
1024	l2r > gr	0.51 [0.468, 0.556]	tie	(✓)	(✗)	(✗)
1024	gr > mg	0.52 [0.474, 0.561]	tie	(✓)	(✓)	(✓)
1024	l2r > mg	0.51 [0.466, 0.553]	tie	(✓)	(✓)	(✓)
512	**gr > l2r**	**0.90 [0.866, 0.919]**	**decided**	✓	✗	✓
512	**mg > l2r**	**0.92 [0.890, 0.938]**	**decided**	✓	✗	✓
512	**mg > gr**	**0.61 [0.564, 0.649]**	**decided**	✓	✓	✓
256	**gr > l2r**	**0.84 [0.805, 0.870]**	**decided**	✓	✗	✓
256	**mg > l2r**	**0.83 [0.792, 0.858]**	**decided**	✓	✗	✓
256	gr > mg	0.54 [0.494, 0.581]	tie	(✗)	(✓)	(✓)
128	**gr > l2r**	**0.67 [0.633, 0.714]**	**decided**	✓	✗	= floor
128	**mg > l2r**	**0.66 [0.621, 0.703]**	**decided**	✓	✗	= floor
128	gr > mg	0.52 [0.473, 0.561]	tie	(✗)	(✗)	(=)
8**B (LLaDA-8B-Base, nucleus** $p = 0.98$**)**						
256 (k=1)	l2r > gr	0.52 [0.472, 0.559]	tie	(✓)	(✗)	(✗)
256 (k=1)	gr > mg	0.53 [0.484, 0.571]	tie	(✓)	(✓)	(✗)
256 (k=1)	l2r > mg	0.54 [0.495, 0.583]	tie	(✓)	(✓)	(✗)
128 (k=2)	**gr > l2r**	**0.71 [0.672, 0.751]**	**decided**	✓	✓	✓
128 (k=2)	**mg > gr**	**0.64 [0.600, 0.684]**	**decided**	✓	✓	✓
128 (k=2)	**mg > l2r**	**0.84 [0.806, 0.870]**	**decided**	✓	✓	✓
64 (k=4)	**gr > l2r**	**0.74 [0.698, 0.775]**	**decided**	✓	✓	✓
64 (k=4)	**mg > l2r**	**0.78 [0.743, 0.815]**	**decided**	✓	✓	✓
64 (k=4)	mg > gr	0.54 [0.496, 0.583]	tie	(✓)	(✓)	(✗)
32 (k=8)	**gr > l2r**	**0.66 [0.619, 0.702]**	**decided**	✓	✓	✓
32 (k=8)	**mg > l2r**	**0.68 [0.633, 0.715]**	**decided**	✓	✓	✓
32 (k=8)	gr > mg	0.50 [0.457, 0.545]	tie	(✗)	(✓)	(✓)

Table 11: **Judged dimensions.** Left: Spearman ρ between DUEL perplexity and the judge's mean score per dimension over each scale's 16 configurations. Right: share of decided-pair verdicts whose reported primary reason is that dimension ($n = 6{,}134$ at 110M, 6,380 at 8B).

	ρ**(dimension, DUEL)**		**% of decided verdicts driven**	
Dimension	110**M**	8**B**	110**M**	8**B**
fluency	−0.93	−0.95	21%	45%
grammaticality	−0.93	−0.95	0%	0%
coherence	−0.93	−0.95	**79%**	**55%**
consistency	−0.92	−0.99	0%	0%
overall score	−0.90	−0.93	permutation $p \approx 10^{-4}$	

Exhibit 1 — entropy collapse rewarded (BD3-LM $L' = 16$**,** NFE $= 128$**).** One left-to-right and one probability-margin generation at the budget where the metrics disagree most; scores in Table 12.

Table 12: **Scores for Exhibits 1 and 2. Bold**: the generation the method ranks better; ✓/✗: agrees/disagrees with the blinded judge; =: identical values. DUEL ppl is the configuration-level value of Table 8; gen-ppl is per sample and MAUVE per configuration.

Exhibit	Method	l2r	Prob. Margin	Agrees?
1: BD3-LM	Pairwise judge (blinded)		**preferred**	ground truth
L'=16, NFE 128	DUEL ppl ↓	240.3	**141.3**	✓
	gen-ppl ↓	**4.76**	336.1	✗
	MAUVE ↑	0.005	0.005	=
2: LLaDA-8B	Pairwise judge (blinded)	**preferred**		ground truth
NFE 256	Absolute judge rating (1–10)	**5**	2	ground truth
	DUEL ppl ↓	**12.37**	12.77	✓
	gen-ppl ↓	34.1	**9.8**	✗
	MAUVE ↑	0.398	**0.427**	✗

Exhibit 1: BD3-LM, block size 16, NFE 128

l2r: "A SHARES pilot with in capital and's slogan message- can accepted every bypass 24 who to this WILL. OUT attempt VIEWAT 2 i JONSorIDIF … IO II, II i A I # AI_II,W, I" I , II I II ,II ISWe IIa I U IN …" — then collapses into "I I I …" repeated ×264 through the end.
prob margin: "A with pilot in a of helicopter landed into Poland's capital, and got in a flight attempt plane17 JAT Warsaw. WarsawAiot described 24, 2015 … Peter condemned dispatch Katow patrol Warsaw … Police " was were duringIN receive a signals overload incident … the Republic of Poland …" — garbled, but a recognizably topical news fragment before degrading.

The single-token loop is extremely *predictable*, so gen-ppl scores it near-minimal, rewarding exactly the failure the judge penalizes.

Exhibit 2 — no entropy collapse, coherence alone (LLaDA-8B, NFE = 256). Both continuations extend the *same* 32-token OWT prefix and neither contains any repetition (longest repeated-token run = 1), so entropy collapse cannot explain the disagreement.

Exhibit 2: LLaDA-8B, NFE 256

prob margin: ", which has the world's highest concentration of volcanoes, was alarmed last week by increased seismic activity under the island … Apart from the movement of tectonic plates and the tremors normally associated with erupting volcanoes, this level of activity is unusual. New Zealand is currently experiencing increasing weekly earthquakes … #Lat News - Earthquake #LatWorld News #[non-Latin gibberish]" — textbook-fluent prose that ends by switching into non-Latin gibberish.
l2r: "has been getting plenty of press for having mild volcanic and ash activity, which has affected flights over Iceland … So I've been thinking about monitoring those landslides, on lookouts in the woods along part of Glacier Road … I am at the Glacier Lake area with friend and colleague Anthony Granko (part owner of Glacier River Sawmill) and his family so I could scrub the area for some more potential landslides …" — rambling, but a coherent first-person narrative held to the end.

The blinded judge picked the l2r sample in *both* presentation orders, citing coherence, and its absolute ratings separate the pair decisively (5/10 against 2/10). gen-ppl nonetheless prefers probability margin by 3.5×: its per-token average is dominated by the smooth first half and forgives the catastrophic tail. With no low-entropy loop to blame, this is gen-ppl's second failure mode — rewarding local smoothness over global coherence.

Judge prompts. The two system prompts, verbatim; the 8B study differs only as noted above.

Absolute prompt (verbatim)

System.

You are evaluating text samples generated by a small (110M-parameter) research language model. Many samples will be partially or severely degraded — disfluent, repetitive, or outright word salad. That is expected: your job is to measure quality, and assigning very low scores to bad text is a correct and useful

outcome, not something to avoid or apologize for. Never refuse to rate a sample because it is low quality or nonsensical.
Rate the sample on four dimensions, each on a 1–10 scale (1 = completely broken, 5 = flawed but partially readable, 10 = indistinguishable from well-edited human text):

- fluency: does the text read naturally word-to-word and sentence-to-sentence?
- grammaticality: is the text syntactically well-formed?
- coherence: do sentences and paragraphs connect logically; does the text stay on topic?
- consistency: are entities, facts, and claims internally consistent across the sample (no contradictions, no entity confusion)?

Then give an overall 1–10 quality score. The overall score is your holistic judgment, not an average.
Notes: the sample may contain several unrelated document fragments separated by a line of dashes ("`---`"); this is an artifact of the generation format — judge each fragment on its own merits and do not penalize the presence of the separator or abrupt topic changes exactly at a separator. Samples may be truncated mid-sentence at the end; do not penalize the final cut-off. Judge only the text between separators.
Respond with a JSON object with integer fields `fluency`, `grammaticality`, `coherence`, `consistency`, and `overall` (each 1-10), plus a brief string field `comment`.

User. "Rate the following text sample." followed by `<sample>{text}</sample>`.

Pairwise prompt (verbatim)

System.

You are comparing two text samples (A and B) generated by small research language models. Either or both may be degraded, repetitive, or nonsensical — that is expected, and picking the less-bad sample is a correct and useful judgment. Never refuse to judge because the text is low quality.
Decide which sample is higher quality overall, weighing fluency, grammaticality, coherence, and internal consistency. Length alone is not quality: do not prefer a sample merely for being longer. Samples may contain unrelated document fragments separated by a line of dashes ("`---`") and may be truncated mid-sentence at the end; penalize neither.
Answer "A" or "B". Answer "tie" only if the two are genuinely indistinguishable in quality — use it sparingly. Also report `primary_reason`: which single dimension most drove your verdict (or "other").
Respond with a JSON object with a string field `winner` ("A", "B", or "tie"), a string field `primary_reason` (one of "fluency", "grammaticality", "coherence", "consistency", "other"), and a brief string field `comment`.

User. "Sample A: `<sample_a>{a}</sample_a>` Sample B: `<sample_b>{b}</sample_b>` Which is higher quality overall?"

E ADDITIONAL EXPERIMENTS

E.1 COST

Each step is one forward pass, so DUEL costs $T = \lceil L/k \rceil$ passes per sequence, against L at $k = 1$; under block decoding with block size L' it is $\lceil L'/k \rceil$ passes per block plus one commit forward on the cached path (Table 13). The boundary case $|\mathcal{M}| < k$ costs a full forward pass for fewer than k tokens, so the measured pass count exceeds L/k slightly whenever $k \nmid L'$. The rest of this subsection measures that cost against the Monte Carlo ELBO on the two workloads of the paper: long held-out sequences, where the ELBO is cheaper, and short benchmark answers, where DUEL is.

Held-out perplexity. Table 13 measures both estimators on the same hardware and the same data: 110M models, OWT, $L = 1024$, batch size 32, one RTX A6000, `sdpa` attention, `fp32` log-probabilities. Per-batch times are means over the interior batches of an eight-batch run; the first batch carries CUDA-context init, cuDNN autotune and lazy compilation and the last is partial, so both are excluded and the exclusion is reported. Per-arm coefficients of variation are 0.2–0.4%, and a calibration arm run on each of nine A6000 nodes spans only $1.043\times$, so cross-node effects move any ratio in the table by at most 2.2%.

Cost model. The ELBO is one forward pass per Monte Carlo sample, at length L for MDLM and SEDD and at length $2L$ for BD3-LM; the M column of Table 13 is the number of *draws per sequence* that pass buys, which is 1 for MDLM and L/L' for BD3-LM. DUEL is

$$\text{forwards} = \underbrace{\tfrac{L}{L'}\left(\lceil L'/k \rceil + 1\right)}_{\text{BD3-LM, length } L' \text{ with a KV prefix}} \quad \text{versus} \quad \underbrace{\tfrac{L}{L'}\lceil L'/k \rceil}_{\text{MDLM/SEDD, length } L},$$

and this asymmetry, not the step count, is what dominates the cost. Two things drive it. First, the $+1$ is a per-block commit forward on the cached path; it is omitted from the naive decoding-step count, is worst relative to the loop at $k = L'$ where it is a flat $2\times$, and is what makes the BD3-LM NFE entries of Table 13 read 1279/1151/1087 rather than 1023 at $k = 1$. Fitting $s = n_{\text{unmask}}u + (L/L')c$ to twelve measurements recovers $u = 89.5$–96.6 ms and $c = 70.8$–77.1 ms with $c/u = 0.79$–0.80 stable across all three block sizes and a maximum fit error of 0.13%; no model without the commit forward fits these points, so the correction is confirmed from timing alone. Second, MDLM cannot use a KV cache at all, so each of its steps recomputes the whole sequence to harvest k tokens. Its timings are exactly linear in the step count (89.950/179.907/359.865 s per batch at $k = 4/2/1$, ratios 2.0001 and 2.0003), confirming that it carries no commit forward and no constant term. Memory follows the same split: DUEL costs +9,422 MiB (+52.5%) of peak GPU memory for MDLM, because the uncached path materializes a $[B, L, V]$ `fp32` log-probability tensor (6.6 GB at $B = 32$, $V = 50{,}258$), against +1,102 to +1,300 MiB (+5.2 to +6.1%) for BD3-LM, whose cached path never allocates at full sequence length. Finally, cost must be quoted per rule and not per NFE: at equal forward count, probability-margin selection costs 1.24–$1.40\times$ block-greedy, since it needs a top-2 over the 50,258-token vocabulary at every masked position. In full-split terms, MDLM's ELBO at MC = 1 costs 0.269 GPU-hours and DUEL at $k = 1$ costs 345 GPU-hours (110,480 sequences = 3,453 batches at $B = 32$).

Caveat on Table 13's perplexities. The cost arms run eight batches, i.e. the first 256 sequences of the unshuffled OWT validation split, which is measurably harder than the split average; the perplexities in Table 13 are therefore higher than the full-split values in Table 1a and should not be read against them. BD3-LM $L' = 16$ calibrates the offset, being the one arm measured on both: 24.39 against 22.27 for the ELBO ($1.095\times$) and 22.20 against 20.76 for DUEL ($1.069\times$). Deflating MDLM's 22.78 by the DUEL factor gives 21.30 against the published 21.86, a 2.6% discrepancy. This is an argument from a calibrated offset, not a measurement: a direct full-split MDLM DUEL run costs the 345 GPU-hours above. What the cost arms do establish directly is that on identical data DUEL beats its own ELBO (22.78 against 24.98), and that the $k = L'$ perplexities sit at the bottom of a monotone ladder in k and are the price of parallel decoding, not a broken path.

Multiple-choice benchmarks. Table 14(a) gives the per-task cost. DUEL spends 18.3–29.4 forwards per scored choice against the ELBO's fixed 128, a 4.4–$7.0\times$ reduction, and is 1.18–$4.48\times$ faster in wall-clock on every task. **Batching is asymmetric and favors the ELBO**: its 128 draws per

choice pack into four batch-32 forwards at full GPU occupancy, while our DUEL implementation scores one sequence at a time, so its far fewer forwards each run on a mostly idle device. *Speedup* (NFE) counts forwards and is a property of the two estimators, portable to any implementation; *speedup* (wall) is what this implementation achieves under that asymmetry and is a **conservative lower bound** — batching DUEL across questions would move it toward the NFE ratio. Both are reported, and the wall-clock advantage holds on every task even without that fix. Table 14(b) prices the same trade on a longer, generation-scored workload, where the scored span is no longer short: on GSM8K at $k = 1$ DUEL costs slightly *more* than the $\mathrm{MC} = 128$ ELBO (9.4 h vs 9.2 h), and raising k coarsens the policy but cuts the cost below it — $k = 2$ halves it and $k = 8$ is 7.7× cheaper than the ELBO. Which side of the crossover a workload falls on is set by L/k against MC, not by the estimator.

Why $\mathrm{MC} = 128$ is the right comparator. The natural objection to Table 3 is that a cheaper, low-MC ELBO would do just as well, since DUEL spends only 18–29 forwards. Table 15 rules this out by rerunning the same benchmarks at every rung of the Monte Carlo ladder, on the full split. At $\mathrm{MC} = 1$ three of the six tasks sit at or below chance (LSAT-LR 16.5 vs 20.0, LSAT-RC 19.0 vs 20.0, LogiQA 24.4 vs 25.0), and accuracy climbs until it plateaus only around $\mathrm{MC} = 64$–128. At $\mathrm{MC} = 32$ — already more forwards than DUEL spends on *any* task — the ELBO is still 1.6 to 12.5 points short of its own $\mathrm{MC} = 128$ accuracy, and 2.3 to 16.3 points below DUEL. Reaching the plateau does not close the difference either: DUEL beats the $\mathrm{MC} = 128$ row on all six tasks while spending 18–29 forwards. $\mathrm{MC} = 128$ is LLaDA's published protocol, not a comparator chosen to flatter DUEL.

E.2 Estimator decomposition

On AG News the ELBO reads 62.21 and DUEL reads 56.50. Is the ELBO a noisy, loose bound, or is it scoring a worse order? BD3-LM lets us answer exactly, because with block size L' all $L'!$ within-block orders can be enumerated.

Setup. The released `bd3lm-owt-block_size4` and `block_size8` checkpoints, evaluated zero-shot on AG News with the GPT-2 tokenizer: 384 sequences of $L = 1024$ with 1023 scored tokens each. Every row of Table 16 scores the *same* checkpoints on the *same* tokens, so rows differ only in the estimator. Entries are corpus perplexities, $\exp(\mathrm{NLL}/L)$; a ladder difference is therefore a difference of perplexities and the differences add along the ladder by construction.

The ladder. Write b for a block of L' positions with all preceding blocks revealed, $\mathcal{S}'$ for the within-block orders ($|\mathcal{S}'| = L'!$), and $p_\theta(\boldsymbol{x} \mid \sigma)$ for the block's likelihood under order σ (Equation (11)); quantities are per block and summed over blocks. The four estimators of $-\log p_\theta(\boldsymbol{x})$ are

$$\text{MC-ELBO} = -\tfrac{1}{n}\textstyle\sum_{i=1}^{n} \log p_\theta(\boldsymbol{x} \mid \sigma^{(i)}), \quad \text{exact uniform-order} = -\tfrac{1}{L'!}\textstyle\sum_{\sigma\in\mathcal{S}'} \log p_\theta(\boldsymbol{x} \mid \sigma),$$
$$\text{DUEL} = -\log p_\theta\big(\boldsymbol{x} \mid \sigma^F(\boldsymbol{x})\big), \qquad \text{order-marginal} = -\log \tfrac{1}{L'!}\textstyle\sum_{\sigma\in\mathcal{S}'} p_\theta(\boldsymbol{x} \mid \sigma),$$

with $\sigma^{(i)} \sim \mathrm{Unif}(\mathcal{S}')$. The MC-ELBO and the exact uniform-order NLL are bounds under π^{unif}; DUEL and the order-marginal are exact likelihoods, of π^F and of π^{unif} respectively. Enumeration costs $L'! \cdot L'$ forward passes per block — 96 at $L'{=}4$ and 322,560 at $L'{=}8$ — which is why the ladder exists only at these two block sizes, and why the order-marginal is a diagnostic rather than a metric.

Reading the three differences. The first difference is noise, not a gap. The exact uniform-order NLL is the ELBO's estimand: across 14 seeds the single-draw ELBO scatters around it with standard deviation 1.18 ppl (mean 60.90 ± 0.32 against 60.72 exact), so 1.49 is a $+1.1$ sd draw and says nothing about the bound. The Jensen gap is an order of magnitude larger (10.50, 13.98) and grows with L', as it must — more within-block orders means more spread in $\log p_\theta(\boldsymbol{x} \mid \sigma)$ for Jensen to exploit. Order selection is the only difference that compares two *policies*: both the exact uniform-order NLL and DUEL are averages of single-order NLLs, so their difference is the choice of order and nothing else. It is rule-dependent, spanning 3.14–6.02 at $L'{=}4$ and 3.64–6.04 at $L'{=}8$ across the three rules, with left-to-right the strongest and probability margin the weakest — a ranking that does not survive into generation quality, where the margin rule wins at low NFE. The last row of

Table 17 is the residual: DUEL sits 6.28 and 9.40 *above* the order-marginal, so it does not tighten the bound toward the marginal; it improves on the practical ELBO through order selection alone. The order-marginal is the better density (50.22 and 48.02, against the AR model's 52.11), but it costs $L'!$ orders per block and, like the ELBO, ignores the test-time policy.

DUEL need not beat the ELBO. Nothing here guarantees DUEL $<$ ELBO. The two numbers score different samplers, $p_\theta^{\pi^F}$ and $p_\theta^{\pi^{\text{unif}}}$, and a deterministic rule that picks a poor order for a corpus is penalized for it; an estimator that could only ever report a lower number than the ELBO would be measuring the bound, not the policy. The closest call in our results, MDLM on Lambada (48.42 against an ELBO draw of 48.29), is within the ELBO's seed-to-seed noise (48.45 ± 0.20 over 12 seeds), so it is a tie rather than a counterexample.

E.3 Sampler selection and downstream accuracy

Section 4.4 validates DUEL's sampler ranking against an LLM judge on open-ended text. Here we ask the more practical question: does the ranking predict which sampler solves a task best? LLaDA-8B-Base answers GSM8K (Cobbe et al., 2021) (1319 test problems, 4-shot) with block size $L' = 32$ under three block samplers, greedy confidence, probability margin and left-to-right, at two budgets, $k = 2$ and $k = 4$ tokens per step. For each cell we record two numbers. The first is pass@1 accuracy of the answers it generates. The second is the DUEL perplexity per answer token of the reference solution given the question, obtained by replaying that sampler's rule on the reference text, so it measures how much probability the sampler's own policy assigns to a correct solution. Neither number needs the other: accuracy needs generations and a grader, DUEL needs only the reference text and one forward pass per step.

Table 18 shows what DUEL gets right and where it stops. At both budgets left-to-right has the highest perplexity and the lowest accuracy, and every deterministic rule beats the ELBO's own random-order sampler by 23 to 31 points while the ELBO, invariant to the sampler, reads 1.70 for all of them. At $k = 2$ the full orderings coincide: probability margin is best on both measures, then greedy confidence, then left-to-right. At $k = 4$ the top pair inverts: DUEL prefers probability margin by 0.10 ppl while greedy confidence is more accurate by 3.1 points (McNemar $p = 0.05$), with left-to-right still last on both.

That inversion is the caveat. DUEL perplexity measures how a policy *scores* correct text when replayed on it; accuracy measures whether the policy *finds* such text when generating, and the two can diverge. Section 4.3 gives another example: scoring answers under left-to-right beats scoring under greedy confidence on most multiple-choice tasks, yet left-to-right is the worst *generator* here. Likelihood and accuracy agree most reliably when the likelihood gap is large, as for left-to-right against the other two rules; for close pairs, whose perplexities differ in the second decimal, the accuracy difference should not be read from the likelihood alone.

Table 13: Cost of likelihood evaluation, 110M models on OWT ($L = 1024$, $B = 32$, one A6000). **NFE/seq** is the true forward count including the block-commit forward; M is the number of ELBO draws one forward pass buys per sequence. Perplexities are on the first 256 sequences of the validation split and are *not* comparable to Table 1a; see below.

Model	Estimator	Setting	NFE/seq	s/seq	PPL	Slowdown
MDLM	ELBO	M=1	1	0.0085	24.98	1×
	DUEL	k=1	1024	11.246	22.78	1328×
	DUEL	k=4 ($= L'$)	256	2.811	116.64	332×
BD3-LM L'=4	ELBO	M=256	1	0.0576	22.76	1×
	DUEL	k=1	1279	3.428	21.09	59.5×
	DUEL	k=4 ($= L'$)	512	1.282	106.10	22.3×
BD3-LM L'=8	ELBO	M=128	1	0.0576	23.95	1×
	DUEL	k=1	1151	3.232	21.79	56.1×
	DUEL	k=8 ($= L'$)	256	0.661	239.21	11.5×
BD3-LM L'=16	ELBO	M=64	1	0.0568	24.39	1×
	DUEL	k=1	1087	3.242	22.20	57.1×
	DUEL	k=16 ($= L'$)	128	0.348	424.30	6.1×

Table 14: Cost. (a) Per-task, LLaDA-8B-Base, one A6000 per arm, both arms back-to-back in one job on a fixed 200–500 document subsample; DUEL accuracy is the greedy (π^F) arm. (b) GSM8K, 1319 sequences, one A6000. NFE per sequence is $\lceil L/k \rceil$ for DUEL and MC for the ELBO, so it does not depend on which rule the policy uses.

(a) per-task cost and accuracy

	DUEL (greedy)		ELBO (MC=128)		speedup		
Task	NFE	s	NFE	s	(NFE)	(wall)	Δ acc
HellaSwag	29.4	1176	128	2141	4.4×	1.82×	+0.7
LSAT-LR	25.2	1877	128	5518	5.1×	2.94×	+1.5
PIQA	22.5	487	128	577	5.7×	1.18×	+1.3
LSAT-RC	21.2	4646	128	20823	6.0×	4.48×	+4.8
LogiQA	20.3	1520	128	5409	6.3×	3.56×	+4.9
MuTual	18.3	1225	128	3955	7.0×	3.23×	+1.5

(b) GSM8K, 1319 sequences

Estimator	NFE/seq	**Cost**
ELBO (MC=128)	MC = 128	9.2 h
DUEL, $k = 1$	$\lceil L/1 \rceil$	9.4 h
DUEL, $k = 2$	$\lceil L/2 \rceil$	4.6 h
DUEL, $k = 8$	$\lceil L/8 \rceil$	1.2 h

Table 15: The Monte Carlo ladder: ELBO accuracy against the number of samples, full split, batch 32. A cheap low-MC ELBO does not suffice. DUEL is the greedy arm.

Estimator	MC / NFE	HellaSwag	LSAT-LR	PIQA	LSAT-RC	LogiQA	MuTual
chance	—	*25.0*	*20.0*	*50.0*	*20.0*	*25.0*	*25.0*
ELBO	1	38.6	16.5	54.2	19.0	24.4	31.4
ELBO	2	45.7	19.0	57.0	23.0	26.4	30.9
ELBO	4	55.2	24.5	61.9	19.3	27.5	37.2
ELBO	8	63.2	19.5	63.4	27.1	30.4	44.6
ELBO	16	67.3	27.5	67.9	39.4	27.6	47.2
ELBO	32	69.1	28.5	71.3	43.9	30.3	48.2
ELBO	64	70.3	39.0	72.9	52.0	34.4	49.5
ELBO	128	70.7	41.0	74.8	55.4	34.1	50.1
DUEL (greedy)	18–29	**71.4**	**42.5**	**76.1**	**60.2**	**39.0**	**51.6**

Table 16: The full ladder at both block sizes, with per-block cost. Same data, same checkpoints, same scored tokens; $n = 1$ per block and $k = 1$. The three middle rows score π^{unif}, the DUEL rows π^{F}; "exact uniform-order" is an exact computation *of a bound*. The MC-ELBO is a single stratified draw scoring all blocks at once; $\pm$ is the standard deviation of that draw across 14 seeds (mean 60.90 ± 0.32), measured at L'=4 only. **Bold**: lowest per column.

Estimator	**Exact?**	**Cost (fwd / block)**	L'=4	L'=8
AR (reference)	exact	1 / sequence (causal)	52.11	52.11
MC-ELBO (one draw)	bound	n / sequence	62.21 ± 1.18	62.97
Exact uniform-order	bound	$L'! \, L'$	60.72	62.00
Order-marginal (mixture)	exact	$L'! \, L'$	**50.22**	**48.02**
DUEL, left-to-right	exact	L'/k	54.70	55.96
DUEL, greedy confidence	exact	L'/k	56.50	57.42
DUEL, probability margin	exact	L'/k	57.58	58.36

Table 17: The three differences the ladder isolates, from Table 16. DUEL rows use greedy confidence; the last row is not a fourth term but the residual it leaves.

Difference	**What it measures**	**Policies**	L'=4	L'=8
MC-ELBO $\rightarrow$ exact uniform-order	sampling noise of one draw (sd 1.18)	π^{unif} vs π^{unif}	1.49	0.97
exact uniform-order $\rightarrow$ order-marginal	Jensen gap	π^{unif} vs π^{unif}	10.50	13.98
exact uniform-order $\rightarrow$ DUEL	order selection	π^{unif} vs π^{F}	4.22	4.58
DUEL $\rightarrow$ order-marginal	DUEL above the marginal	π^{F} vs π^{unif}	6.28	9.40

Table 18: **DUEL perplexity against downstream accuracy.** LLaDA-8B-Base on GSM8K (Cobbe et al., 2021), block size $L' = 32$, $n = 1319$; perplexity per answer token of the reference solution, accuracy pass@1. The ELBO row is its Monte Carlo estimate (128 samples), identical for every sampler, paired with the accuracy of its own random-order sampler. **Bold**: best per column among the three samplers.

	$k = 2$ (NFE $= 128$)		$k = 4$ (NFE $= 64$)	
Sampler	DUEL ppl $\downarrow$	acc. $\uparrow$	DUEL ppl $\downarrow$	acc. $\uparrow$
Block greedy confidence	1.73	64.0%	2.10	**51.9%**
Block probability margin	**1.72**	**64.1%**	**2.00**	48.8%
Block left-to-right	1.79	61.6%	2.38	43.5%
ELBO (random-order sampler)	1.70	37.8%	1.70	20.7%

F REPRODUCIBILITY

We train nothing. Every checkpoint in this paper is a public release, evaluated unchanged: the 110M SEDD, MDLM, BD3-LM and AR baselines of Lou et al. (2024); Sahoo et al. (2024); Arriola et al. (2025a), the SMDM IsoFLOP grids of Nie et al. (2024), and LLaDA-8B-Base (Nie et al., 2025). DUEL needs no retraining, so reproducing any number here means downloading the checkpoint and running the evaluation protocol of Appendix D with the settings in Table 19. Training budgets in that table are quoted from the releases for comparability; the 110M arms share one backbone and schedule and differ only in objective, which is what makes their comparison like-for-like. The SMDM grid is scored on 800 held-out SlimPajama sequences per checkpoint, with the ELBO and DUEL scoring identical sequences so that their difference is paired.

Hardware and numerics. Each arm runs on one GPU. The arms whose timings appear in Appendix E.1 were pinned to a single RTX A6000 with `sdpa` attention and `fp32` log-probabilities, so that their ratios are comparable; a calibration arm run across nine A6000 nodes spans $1.043\times$, so cross-node effects move any reported ratio by at most 2.2%. The remaining arms, the generation runs behind Section 4.4, the SMDM sweep and the 8B accuracy runs, were scheduled by Slurm on whichever of RTX A6000, RTX 6000 Ada, A100, H100 or H200 was free, which changes wall-clock but not a likelihood beyond floating-point rounding. Those arms were not timed end to end, and no project-wide total including preliminary and abandoned runs is recoverable from the logs we hold.

Code. The DUEL implementation, the configuration of every arm in this paper, and the Slurm scripts behind the timings of Appendix E.1 will be released upon acceptance.

Table 19: Checkpoints, training budgets (from their releases) and the block size each is evaluated at. The 110M rows share one backbone, a 12-layer DiT of width 768 with 12 heads, causal for the AR baseline; the two L values are OWT and LM1B. SMDM's five IsoFLOP budgets are $C \in \{6, 10, 30, 60, 100\} \times 10^{18}$ FLOPs under its own $C = 6ND$ convention. L' is native for BD3-LM and an evaluation choice for SEDD/MDLM/LLaDA, which are trained globally. All DUEL runs in the paper use $k = 1$ except the k-ladders of Appendix E.1, the budget sweeps of Appendix D.4 and the GSM8K runs of Appendix E.3.

Model	**Params**	**Training budget**	L	L' (eval)
AR baseline	110M	1M steps $\times$ 512, causal	1024 / 128	—
SEDD	110M	1M steps $\times$ 512	1024 / 128	4
MDLM	110M	1M steps $\times$ 512	1024 / 128	4
BD3-LM	110M	1M steps $\times$ 512	1024 / 128	4, 8, 16 (native)
SMDM grid	19M–1028M	five IsoFLOP budgets, $C = 6ND$	2048	32
LLaDA-8B-Base	8B	as released	2048	32