LEARNING GENERATION ORDERS FOR MASKED DISCRETE DIFFUSION MODELS VIA VARIATIONAL INFERENCE

David Fox *
University of Bristol
david.k.fox@bristol.ac.uk

Sam Bowyer
University of Bristol
sam.bowyer@bristol.ac.uk

Song Liu
University of Bristol
song.liu@bristol.ac.uk

Laurence Aitchison
Mistral AI
laurence.aitchison@gmail.com

Raul Santos-Rodriguez
University of Bristol
enrsr@bristol.ac.uk

Mengyue Yang
University of Bristol
mengyue.yang@bristol.ac.uk

ABSTRACT

Masked discrete diffusion models (MDMs) are a promising new approach to generative modelling, offering the ability for parallel token generation and therefore greater efficiency than autoregressive counterparts. However, achieving an optimal balance between parallel generation and sample quality remains an open problem. Current approaches primarily address this issue through fixed, heuristic parallel sampling methods. There exist some recent learning based approaches to this problem, but its formulation from the perspective of variational inference remains underexplored. In this work, we propose a variational inference framework for learning parallel generation orders for MDMs. As part of our method, we propose a parameterisation for the approximate posterior of generation orders which facilitates parallelism and efficient sampling during training. Using this method, we conduct preliminary experiments on the GSM8K dataset, where our method performs competitively against heuristic sampling strategies in the regime of highly parallel generation. For example, our method achieves 33.1% accuracy with an average of only 4 generation steps, compared to 23.7-29.0% accuracy achieved by standard competitor methods in the same number of steps. We believe further experiments and analysis of the method will yield valuable insights into the problem of parallel generation with MDMs.

1 INTRODUCTION

Discrete Diffusion Models (DDMs) Austin et al. (2021) have recently emerged as a promising alternative to autoregressive generative models for various tasks such as text Song et al. (2025), code Khanna et al. (2025) and biological sequence generation Vignac et al. (2022), offering the ability to parallelise token generation for increased efficiency Hu et al. (2025) and utilise bi-directional context during generation which has shown to be beneficial in certain tasks where a strict left-to-right dependency is not present in the data Nie et al. (2025b). Masked Diffusion Models (MDMs) Austin et al. (2021) in particular have established themselves as the most performant class of DDMs, due to their comparatively efficient paramaterisation and conceptually simple set-up Shi et al. (2024); Sahoo et al. (2024), as well as their similarity with Autoregressive Models (ARMs), making it possible to initialise training from strong autoregressive baselines Ye et al. (2025).

*Corresponding author.

Despite the conceptual benefits and recent empirical progress of DDMs Nie et al. (2025a), a consistent barrier to realising their full potential has been optimally balancing efficiency and sample quality by choosing generation orders which use parallelization without violating statistical dependence among token positions Kang et al. (2025).

Current approaches attempt to overcome this issue by using heuristic sampling strategies, or by augmenting the generative model with a learned component which chooses token positions to unmask at each step. Heuristic approaches typically use model predicted logits to select tokens based on criteria such as top-k or top probability margin Kim et al. (2025). Learned approaches usually augment the generative model to include a component for selecting token indices which is then trained with a separate loss function Zhangzhi Peng et al. (2025), or via reinforcement learning by formulating the generative model as a Markov Decision Process with a policy that chooses token positions to unmask Hong et al. (2025).

While these methods have made progress toward achieving the optimal trade-off between generation efficiency and model output quality, we believe there is room for improvement. While heuristic approaches provide good performance at minimal cost, they are potentially too rigid and overly reliant on logit based model confidence estimates, which my be poorly calibrated for this purpose when trained using only binary cross-entropy. Of the learned approaches, the formulation of learned parallel generation orders within the variational inference framework remains underexplored.

Hence, the goal of the present work is to investigate a variational inference framework for training an MDM which explicitly factorises the model into components for choosing which token positions to unmask, and which token value to sample given a position. We believe this formulation offers the following benefits which may lead to a method that allows training of an improved unmasking position model which scales to large datasets. Our main contributions are as follows:

- We present a probabilistic formulation of a discrete diffusion generative model using variational inference, which explicitly factorises the model into components for choosing which token positions to unmask, and which token value to sample given a position.
- We derive the associated ELBO objective, which leverages this model structure to decrease variance of the objective function through Rao-Blackwellisation.
- We investigate the use of a paramaterised family of distributions for the approximate posterior generation order which is designed to allow efficient, low variance training.

The remainder of this document is structured as follows: In section 2 we discuss necessary background information on masked DDMs, the issue of accurately sampling in parallel, and some current approaches to overcoming this issue. In section 3 we discuss our method, which is motivated by these concerns, and a plan for experiments to test it, which are reported in section 4 and discussed in section 5.

2 Background

2.1 Masked Discrete Diffusion Models

To construct an MDM Shi et al. (2024) for a single dimension, we define a forward Markov process $\{x_t\}_{t\in[0,1]}$, taking values in a discrete set $x_t \in \{1, \dots, m-1, m\}$, where m represents a special 'mask' token. Letting $\mathbf{1}$ denote a vector of ones and $\mathbf{x}_s$ the one hot encoding of x_s, the forward process transition probabilities between times $s < t$ are defined as

$$\begin{cases} q_{t|s}(x_t|x_s) = \text{Cat}(x_t; \bar{Q}(s,t)^T \mathbf{x}_s), \\ \bar{Q}(s,t) = \dfrac{\alpha_t}{\alpha_s} I + \left(1 - \dfrac{\alpha_t}{\alpha_s}\right) \mathbf{1}\mathbf{e}_m^T, \end{cases} \tag{1}$$

and the initial distribution at $t = 0$ is the data distribution. The generative model is then defined as an approximation of the time reversal of the forward process. Paramaterising this requires the reverse time transition probability conditional on a clean data sample x_0,

$$\begin{cases} q_{s|t,0}(x_s|x_t, x_0) = \text{Cat}(x_s; \bar{R}(t,s)^T \mathbf{x}_t), \\ \bar{R}^{x_0}(t,s) = I + \dfrac{\alpha_s - \alpha_t}{1 - \alpha_t} \mathbf{e}_m \left(\mathbf{x}_0 - \mathbf{e}_m\right)^T, \end{cases} \tag{2}$$

which is combined with a learned approximation μ_θ of $q_{0|t}$ to give the reverse transition probability,

$$p_\theta\left(x_s|x_t\right) = q_{s|t,0}\left(x_s|x_t, \mu_\theta(x_t,t)\right). \tag{3}$$

To extend to the multi-dimensional case x_t = $x_t^1 ... x_t^N$ without sacrificing computational tractability, we use the following conditional independence assumptions,

$$\begin{cases} q_{t|s}(x_t|x_s) = \displaystyle\prod_{n=1}^{N} q_{t|s}\left(x_t^n|x_s^n\right), \\ q_{s|t,0}\left(x_s|x_t, x_0\right) = \displaystyle\prod_{n=1}^{N} q_{s|t,0}\left(x_s^n|x_t^n, x_0^n\right), \\ p_\theta\left(x_s|x_t\right) = \displaystyle\prod_{n=1}^{N} q_{s|t,0}\left(x_s^n|x_t^n, \mu_\theta^n(x_t,t)\right), \end{cases} \tag{4}$$

where $\mu_\theta^n(x_t,t)$ is trained to approximate $q_{0|t}(x_0^n|x_t)$. The model weights θ are trained by minimising the ELBO for the model likelihood, using the forward model Q as a fixed variational posterior.

We can generate samples from the model via ancestral sampling using p_θ at discrete time points. Specifically, we discretise $[0,1]$ uniformly into T points, $\{t/T\}_{t=0}^T$, initialise $\mathbf{x}_1 = m...m$, and iteratively sample the $\mathbf{x}_t$'s according to $p_\theta\left(x_s|x_t\right)$ [1]

$$\begin{aligned} \mathbf{x}_t &\sim \prod_{n=1}^{N} q\left(x_t^n \middle| x_{t+1}^n, \mu_\theta^n(\mathbf{x}_{t+1}, t)\right), \quad t = T-1, ..., 0 \\ \mathbf{x}_T &= (m...m) \end{aligned}$$

2.2 Reparamaterised Discrete Diffusion

As noted in previous work Zheng et al. (2023), this time discretised model can be reparamaterised to explicitly include i.i.d binary token selection variables r_t^n, in a way that preserves the marginals of $\mathbf{x}_{0:T}$, as follows

$$P_\theta(\mathbf{x}_{0:T}, \mathbf{r}_{0:T-1}) = \delta_{(m...m)}(\mathbf{x}_T) \prod_{t=0}^{T-1} \left[\prod_{n=1}^{N} P_\theta(x_t^n|r_t^n, \mathbf{x}_{t+1})\right] \left[\prod_{n=1}^{N} P(r_t^n|\mathbf{x}_t)\right] \tag{5}$$

where

$$\begin{aligned} P(r_t^n|\mathbf{x}_t) &= \text{Bern}\left(\frac{\alpha_t - \alpha_{t+1}}{1 - \alpha_{t+1}} \mathbf{1}_{x_{t+1}^n = m}\right), \\ P_\theta(x_t^n|r_t^n, \mathbf{x}_{t+1}) &= \delta_{x_{t+1}^n}(1 - r_t^n) + \mu_\theta^n(\mathbf{x}_{t+1}, t) r_t^n. \end{aligned}$$

This paramaterisation is useful as the explicit separation between components allows us to better understand the importance of the method we choose to select token positions to unmask. Notably, the authors of Ben-Hamu et al. (2025) use this form to show that the error between the ground truth

[1] We use the shorthand $x_t = x_{t/T}$

and generative model distributions can be decomposed into a term for error in the denoising network and the mutual information between simultaneously generated token position distributions.

Furthermore, this decomposition provides a template for more generally paramaterising a learnable token selection component. Indeed, given that DDMs are already trained via ELBO optimisation by considering $\mathbf{x}_{1:T}$ as latent variable models, a natural extension is to follow the same procedure, but considering $\mathbf{r}_{0:T-1}$ as additional latent variables. Similar approaches have been applied successfully to learning generation orders for any-order autoregressive models Wang et al. (2025).

3 Learning Generation Orders for Discrete Diffusion Models

We formulate training an MDM with a learned token selection component as variational inference of a latent variable model through ELBO optimisation.

3.1 Generative Model & Approximate Posterior

The generative model has a similar structure to equation 5,

$$P_\theta(\mathbf{x}_{0:T}, \mathbf{r}_{0:T-1}) = \delta_{(m\ldots m)}(\mathbf{x}_T) \prod_{t=0}^{T-1} \left[\prod_{n=1}^{N} P_\theta(x_t^n | r_t^n, \mathbf{x}_{t+1})\right] \left[\prod_{n=1}^{N} P_\psi(r_t^n | \mathbf{x}_t)\right], \tag{6}$$

but with a learned distribution for the unmasking variables r_t^n incorporated,

$$P_\Psi(r_t^n | \mathbf{x}_{t+1}) = \text{Bern}\left(p_\psi^{t,n}(\mathbf{x}_{t+1}) \mathbf{1}_{x_{t+1}^n = m}\right). \tag{7}$$

The approximate posterior used for variational inference has a similar form to that of the generative model,

$$Q_\phi(\mathbf{x}_{0:T}, \mathbf{r}_{0:T}) = \delta_{(m\ldots m)}(\mathbf{x}_T) \prod_{t=1}^{T-1} \left[\prod_{n=1}^{N} Q(x_t^n | r_t^n, x_{t+1}^n, x_0^n)\right] \left[\prod_{n=1}^{N} Q_\phi(r_t^n | \mathbf{x}_{t+1}, \mathbf{x}_0)\right], \tag{8}$$

where

$$Q(x_t^n | r_t^n, x_{t+1}^n, x_0^n) = \delta_{x_{t+1}^n}(1 - r_t^n) + \delta_{x_0^n} r_t^n, \tag{9}$$

$$Q_\phi(r_t^n | \mathbf{x}_{t+1}, \mathbf{x}_0) = \text{Bern}\left(q_\phi^{t,n}(\mathbf{x}_{t+1}, \mathbf{x}_0) \mathbf{1}_{x_{t+1}^n = m}\right). \tag{10}$$

We choose to parameterize $\mathbf{r}_t | \mathbf{x}_{t+1}, \mathbf{x}_0$ and $\mathbf{r}_t | \mathbf{x}_{t+1}$ as vectors of iid Bernoulli random variables as this will allow us to analytically compute certain expectations in the ELBO, leading to a lower variance objective. This can be seen more clearly in the next section.

3.2 Loss Function

We train the model by maximising the ELBO, which for a single data point $\mathbf{x}_0$ is defined as

$$\ln P_{\theta,\psi}(\mathbf{x}_0) \geq \mathbb{E}_{Q_\phi(\mathbf{x}_{1:T}, \mathbf{r}_{0:T} | \mathbf{x}_0)} \left[\log\left(\frac{P_{\theta,\psi}(\mathbf{x}_0, \mathbf{x}_{1:T}, \mathbf{r}_{0:T})}{Q_\phi(\mathbf{x}_{1:T}, \mathbf{r}_{0:T} | \mathbf{x}_0)}\right)\right] = \mathcal{L}(\mathbf{x}_0, \theta, \psi, \phi). \tag{11}$$

Using the properties of conditional independence across timesteps and data dimensions, as well as the forms of the distributions in equations 6 and 8, we can write the ELBO as

$$\mathcal{L}(\mathbf{x}_0, \theta, \psi, \phi) = (T-1) \mathbb{E}_{\substack{Q_\phi(\mathbf{x}_{\geq t+1} | \mathbf{x}_0) \\ t \sim \text{Unif}(0, T-1)}} [L_t] \tag{12}$$

where

$$L_t = \mathbb{E}_{Q_\phi(\mathbf{x}_{\geq t+1}, \mathbf{r}_{\geq t+1}|\mathbf{x}_0)} \Bigg[\sum_{\substack{n=1 \\ \text{s.t. } x^n_{t+1}=m}}^{N} q_\phi^{t,n}(\mathbf{x}_{t+1}, \mathbf{x}_0) \log \mu_\theta^n(\mathbf{x}_{t+1})_{x_0^n} - D_{KL}\left(Q_\phi(r_t^n|\mathbf{x}_{t+1}, \mathbf{x}_0)||P_\psi(r_t^n|\mathbf{x}_{t+1})\right) \Bigg]. \tag{13}$$

See appendix A for a derivation of this result. The loss function reveals the multiple purposes fulfilled by the posterior probabilities $q_\phi^{t,n}$. It determines the partially masked sequences that the denoiser and token position selector see during training, as well as weight the denoiser cross-entropy loss at each masked token position in proportion to its probability of being unmasked at the current timestep, and provide unmasking probabilities for the token selector to match.

In terms of the effect the denoiser and token selector have on $q_\phi^{t,n}$, we can see the first term in the expectation of equation (10) encourages Q to learn unmasking orders which maximise the denoiser confidence in the ground truth tokens, by leveraging information about x_0. The KL-divergence term encourages Q to maintain an unmasking schedule which can be replicated by the token selector P_ψ used during inference, so that there is no mismatch between generation orders seen during training and inference.

Due to the presence of learned parameters ψ in the distribution Q, we use REINFORCE to obtain an unbiased estimate of gradients of the loss. In order to reduce excessive variance of this estimator, we use REINFORCE-Leave-One-Out (RLOO) control variates Kool et al. (2019). Details of gradient estimation are in appendix B.

3.3 Variational Posterior Design

Up until this point, the form of the posterior unmasking probabilities $q^{t,n}$ has remained abstract. In this section, we describe the design choices explored in this work. First, we state some desirable properties that we want the posterior to satisfy:

1. Sampling up to a given time $t \in \{0, ..., T-1\}$ should be computationally efficient (i.e., not scale significantly with t), so that we can easily compute unbiased Monte Carlo estimates of the loss at a randomly sampled time point during training,
2. The posterior should be capable of parallel generation,
3. The posterior should encode a notion of generation order i.e., that certain tokens should be generated before others.
4. The posterior should unmask at least one token in each sampling step, in order to avoid wasted computation during training and inference

To satisfy these properties, we compute the posterior unmasking probabilities $q_\phi^{t,n}(\mathbf{x}_{t+1}, \mathbf{x}_0)$ through a sequence of lightweight re-normalisation steps, based on an initial sequence of scores computed by a neural network $\alpha : \mathcal{V}^N \to [0, 1]^N$ with learnable paramaters ϕ,

$$q_\phi^{t,n}(\mathbf{x}_{t+1}, \mathbf{x}_0) = e^{\left(\alpha(\mathbf{x}_0)_n - \text{Max}\{\alpha(\mathbf{x}_0)_n | n \text{ s.t. } x^n_{t+1}=m\}\right)/\tau}. \tag{14}$$

Note that this design choice satisfies property 1 as sampling requires a single pass through a neural network, followed by a sequence of t updates of negligible cost. It satisfies property 2 as indices with similar α values have a high probability of being generated in the same step. Property 3 is satisfied by assigning higher α scores to tokens which should be unmasked earlier. Finally, property 4 is satisfied by the Max normalisation ensuring that at least one token is unmasked with probability 1 in each step. Empirically, we have found the inclusion of the temperature scaling parameter to be beneficial, typically setting it between 0.1 and 0.05 in our experiments. We hypothesise this may be due to one or both of the following reasons; 1) the temperature scaling prevents the randomly

Table 1: **GSM8K Performance.** Comparison of our learned unmasking order against the baseline at equivalent average and maximum step costs. Since for $T = 10$ our method's average number of steps rounds up to its maximum number of steps, we report the performance of baselines at the floor of this average. Bold values in the final column represent maximum accuracy at the average number of steps that our method took after being trained with a given budget.

Budget (T)	**Method**	**Sampling Cost**	**Avg. Steps**	**Range**	**Acc. (%)**
5	IID	ID @ Avg.	4.0	[4, 4]	29.0
	IID	IID @ Max Used	5.0	[5, 5]	29.9
	Top Prob	Top Prob @ Avg.	4.0	[4, 4]	23.7
	Top Prob	Top Prob @ Max Used	5.0	[5, 5]	26.6
	Top Prob Marg.	Top Prob Marg. @ Avg.	4.0	[4, 4]	24.0
	Top Prob Marg.	Top Prob Marg. @ Max Used	5.0	[5, 5]	27.0
	Ours	**Learned Order**	**4.01**	**[2, 5]**	**33.1**
10	IID	ID @ $\lfloor$Avg.$\rfloor$	9.0	[9, 9]	34.2
	IID	IID @ Max Used	10.0	[10, 10]	36.0
	Top Prob	Top Prob @ $\lfloor$Avg.$\rfloor$	9.0	[9, 9]	35.9
	Top Prob	Top Prob @ Max Used	10.0	[10, 10]	37.8
	Top Prob Marg.	Top Prob Marg. @ $\lfloor$Avg.$\rfloor$	9.0	[9, 9]	36.9
	Top Prob Marg.	Top Prob Marg. @ Max Used	10.0	[10, 10]	**39.5**
	Ours	**Learned Order**	**9.57**	**[7, 10]**	37.8
15	IID	ID @ Avg.	9.0	[9, 9]	34.2
	IID	IID @ Max Used	12.0	[12, 12]	37.0
	Top Prob	Top Prob @ Avg.	9.0	[9, 9]	35.9
	Top Prob	Top Prob @ Max Used	12.0	[12, 12]	41.1
	Top Prob Marg.	Top Prob Marg. @ Avg.	9.0	[9, 9]	36.9
	Top Prob Marg.	Top Prob Marg. @ Max Used	12.0	[12, 12]	42.3
	Ours	**Learned Order**	**9.43**	**[5, 12]**	**39.0**

initialised α scores used at the start of training from unmasking in too few steps, leading to the majority of samples in a batch containing no training signal, 2) temperature scaling decreases the stochasticity in the unmasking orders generated by Q, leading to lower variance training and faster convergence.

4 Experiments

We conduct experiments on the GSM8k dataset, initially following the same supervised finetuning method as Shin et. al. Nie et al. (2025a) on a 170M parameter MDM for 45,000 steps with batch size 256, before carrying out further training with our algorithm using this pre-trained denoiser, and randomly initialised networks for α and $p_\psi^{t,n}$. Specifically, we train according to the method laid out in Section 3 for an additional 15,000 steps using a batch size of 32 and drawing 8 RLOO generations per dataset sample. As a baseline for comparison, we continue finetuning the vanilla 170M MDM (without α and $p_\psi^{t,n}$ networks) on GSM8k, but with a batch size of 256 for 15,000 steps. This results in both our model and the baseline model seeing a batchsize of 256 samples, however, the baseline does get to see more diverse batches.

To decode on the baseline model, we use a linear unmasking schedule to control the number of decoding steps and consider three standard sampling strategies, the first being the simplest approach from Nie et al. (2025a), with the second and third suggested by Kim et al. (2025).

- **IID**: Every masked token is unmasked independently with the same probability, determined by the linear unmasking schedule.
- **Top Probability**: At each decoding step, the number of tokens to be unmasked, K, is determined by the linear unmasking schedule. Then we unmask at the K indices for which

our denoiser has maximal confidence about which token in the vocabulary, $\mathcal{V}$, to predict. That is, we unmask indices i in

$$\underset{i}{\text{Top-k}}(\max \mu_\theta^i(\mathbf{x}_{t+1}, t)).$$

- **Top Probability Margin**: Similarly to **Top Probability**, we unmask K masked tokens, but instead of choosing those with the highest confidence, we choose those with the largest difference between the probability assigned to the *most likely* token, $j_1 \in \mathcal{V}$, and the *second most likely* token, $j_2 \in \mathcal{V}$. That is, we unmask at indices i in

$$\underset{i}{\text{Top-k}} \left(\mu_\theta^i(\mathbf{x}_{t+1}, t)_{j_1} - \mu_\theta^i(\mathbf{x}_{t+1}, t)_{j_2}\right).$$

  This method attempts to improve upon **Top Probability** by not unmasking a token until there is a single most likely value for it to take, avoiding situations where multiple token-values have high-probability.

We compare the results of our method for multiple training budgets of T (the maximum number of decoding steps). Since our method performs *adaptive unmasking*, in the sense that it does not always use the same number of decoding steps for every prompt, we report the mean, min and max number of decoding steps on the GSM8k test set, and compare against the baseline methods evaluated at the mean and max values.

5 Discussion

As can be seen in Table 1, our method successfully learns generation strategies that outperform the baseline generation methods at comparable or higher numbers of steps. This suggests that our method is indeed performing parallel generation in a manner that better avoids the pitfalls of over-parallelisation, especially when we consider the extremely low-budget T=5 setting.

One exception is the case where our model trained with a budget of $T = 10$ ends up using an average of 9.57 steps and achieves slightly worse performance than top probability margin sampling with 10 steps (but better than the baselines with 9 steps). Clearly the extra 0.43 steps give the top probability margin method a slight advantage, and so a direct comparison is difficult, but this does line up with the observation that the gap in performance between our method and the non-IID baselines closes as the decoding budget increases and the risk of over-parallelisation error decreases.

Whilst our results in Table 1 demonstrate a valid proof-of-concept for our method, a full analysis would require many more experiments in future work. For example, we would like to analyse the performance of our method on more datasets and with MDMs of varying sizes. In the development of this paper, we experimented with a variety of approximate posterior forms before ending up with Eq. 14, which seems to perform well. Further experimentation of approximate posterior forms could be a fruitful direction for future work.

6 Conclusion

In this work, we present a method by which discrete diffusion models can decide the generation order of tokens at inference time, using a small learned auxiliary network. This allows the model to adaptively adjust its degree of parallelism based on the task at hand: whilst parallel generation is one of the main advantages of DDMs compared to standard ARMs, generation schemes which lead to too much parallelism can hurt downstream task performance (Kang et al., 2025). We explore a strategy for learning this auxiliary network via variational inference, wherein we treat the generation order as a latent variable to be inferred, and show that this method achieves competitive results against standard baseline MDM generation schemes on GSM8k. We anticipate that further analysis and development of this method would be very helpful towards improving the performance of discrete diffusion models at large.

ACKNOWLEDGMENTS

The authors wish to acknowledge and thank the financial support of the UK Research and Innovation (UKRI) [Grant ref EP/Y030796/1] and the University of Bristol. Sam Bowyer is supported by the UKRI EPSRC via the COMPASS CDT at the University of Bristol (EPS0235691). This work was possible thanks to the computational facilities of the University of Bristol's Advanced Computing Research Centre—http://www.bris.ac.uk/acrc/. We would like to thank Dr.Stewart for funding compute resources used in this project.

REFERENCES

Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in neural information processing systems*, 34:17981–17993, 2021.

Heli Ben-Hamu, Itai Gat, Daniel Severo, Niklas Nolte, and Brian Karrer. Accelerated Sampling from Masked Diffusion Models via Entropy Bounded Unmasking, May 2025.

Chunsan Hong, Seonho An, Min-Soo Kim, and Jong Chul Ye. Improving discrete diffusion unmasking policies beyond explicit reference policies. *arXiv preprint arXiv:2510.05725*, 2025.

Zhanqiu Hu, Jian Meng, Yash Akhauri, Mohamed S Abdelfattah, Jae-sun Seo, Zhiru Zhang, and Udit Gupta. Accelerating diffusion language model inference via efficient kv caching and guided diffusion. *arXiv preprint arXiv:2505.21467*, 2025.

Wonjun Kang, Kevin Galim, Seunghyuk Oh, Minjae Lee, Yuchen Zeng, Shuibai Zhang, Coleman Hooper, Yuezhou Hu, Hyung Il Koo, Nam Ik Cho, et al. Parallelbench: Understanding the tradeoffs of parallel decoding in diffusion llms. *arXiv preprint arXiv:2510.04767*, 2025.

Samar Khanna, Siddhant Kharbanda, Shufan Li, Harshit Varma, Eric Wang, Sawyer Birnbaum, Ziyang Luo, Yanis Miraoui, Akash Palrecha, Stefano Ermon, et al. Mercury: Ultra-fast language models based on diffusion. *arXiv preprint arXiv:2506.17298*, 1, 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, August 2025.

Wouter Kool, Herke van Hoof, and Max Welling. Buy 4 reinforce samples, get a baseline for free! 2019.

Shen Nie, Fengqi Zhu, Chao Du, Tianyu Pang, Qian Liu, Guangtao Zeng, Min Lin, and Chongxuan Li. Scaling up Masked Diffusion Models on Text, February 2025a. URL `http://arxiv.org/abs/2410.18514`. arXiv:2410.18514 [cs].

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*, 2025b.

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.

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

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

Clement Vignac, Igor Krawczuk, Antoine Siraudin, Bohan Wang, Volkan Cevher, and Pascal Frossard. Digress: Discrete denoising diffusion for graph generation. *arXiv preprint arXiv:2209.14734*, 2022.

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*, 2025.

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.

Fred Zhangzhi Peng, Zachary Bezemek, Sawan Patel, Jarrid Rector-Brooks, Sherwood Yao, Alexander Tong, and Pranam Chatterjee. Path planning for masked diffusion model sampling. *arXiv e-prints*, pp. arXiv–2502, 2025.

Lin Zheng, Jianbo Yuan, Lei Yu, and Lingpeng Kong. A reparameterized discrete diffusion model for text generation. *arXiv preprint arXiv:2302.05737*, 2023.

A Appendix

A.1 Masked Diffusion Loss Function Derivation

The generative model is trained using the ELBO of $P_{\theta,\psi}$, using amortised posterior Q_ϕ,

$$\mathcal{L}(\mathbf{x}_0,\theta,\psi,\phi) = \mathbb{E}_{Q_\phi(\mathbf{x}_{1:T},\mathbf{r}_{0:T}|\mathbf{x}_0)}\left[\log\left(\frac{P_{\theta,\psi}(\mathbf{x}_0,\mathbf{x}_{1:T},\mathbf{r}_{0:T})}{Q_\phi(\mathbf{x}_{1:T},\mathbf{r}_{0:T}|\mathbf{x}_0)}\right)\right]. \tag{15}$$

In what follows we use the notation $\mathbf{x}_{\geq t} = \mathbf{x}_{t:T-1}$.

Applying conditional independence over timesteps t gives,

$$\begin{aligned}
\mathcal{L}(\mathbf{x}_0,\theta,\psi,\phi) &= \mathbb{E}_{Q_\phi(\mathbf{x}_{\geq 1},\mathbf{r}_{\geq 0:}|\mathbf{x}_0)}\left[\log P_{\theta,\psi}(\mathbf{x}_0,\mathbf{r}_0|\mathbf{x}_1)\right] + \mathbb{E}_{Q_\phi(\mathbf{x}_{\geq 1},\mathbf{r}_{\geq 0}|\mathbf{x}_0)}\left[\log\prod_{t=1}^{T-1}\frac{P_{\theta,\psi}(\mathbf{x}_t,\mathbf{r}_t|\mathbf{x}_{t+1})}{Q_\phi(\mathbf{x}_t,\mathbf{r}_t|\mathbf{x}_{t+1},\mathbf{x}_0)}\right] \\
&= \mathbb{E}_{Q_\phi(\mathbf{x}_{\geq 1},\mathbf{r}_{\geq 0}|x_0)}\left[\log P_{\theta,\psi}(\mathbf{x}_0,\mathbf{r}_0|\mathbf{x}_1)\right] + \sum_{t=1}^{T-1}\mathbb{E}_{Q_\phi(\mathbf{x}_{\geq t},\mathbf{r}_{\geq t}|\mathbf{x}_0)}\left[\log\frac{P_{\theta,\psi}(\mathbf{x}_t,\mathbf{r}_t|\mathbf{x}_{t+1})}{Q_\phi(\mathbf{x}_t,\mathbf{r}_t|\mathbf{x}_{t+1},\mathbf{x}_0)}\right] \\
&= L_0 + \sum_{t=1}^{T-1} L_t.
\end{aligned}$$

Applying the factorization over $\mathbf{x}_t, \mathbf{r}_t$ in P and Q to the L_t terms gives

$$\begin{aligned}
L_t = \mathbb{E}_{Q_\phi(\mathbf{x}_{\geq t+1},\mathbf{r}_{\geq t+1}|\mathbf{x}_0)}&\left[\mathbb{E}_{Q(\mathbf{x}_t|\mathbf{r}_t,\mathbf{x}_{t+1},\mathbf{x}_0)Q_\phi(\mathbf{r}_t|\mathbf{x}_{t+1},\mathbf{x}_0)}\left[\log\frac{P_\theta(\mathbf{x}_t|\mathbf{r}_t,\mathbf{x}_{t+1})}{Q(\mathbf{x}_t|\mathbf{r}_t,\mathbf{x}_{t+1},\mathbf{x}_0)}\right]\right. \\
&\left.+\mathbb{E}_{Q(\mathbf{x}_t|\mathbf{r}_t,\mathbf{x}_{t+1},\mathbf{x}_0)Q_\phi(\mathbf{r}_t|\mathbf{x}_{t+1},\mathbf{x}_0)}\left[\log\frac{P_\psi(\mathbf{r}_t|\mathbf{x}_{t+1})}{Q_\phi(\mathbf{r}_t|\mathbf{x}_{t+1},\mathbf{x}_0)}\right]\right] \\
= \mathbb{E}_{Q_\phi(\mathbf{x}_{\geq t+1},\mathbf{r}_{\geq t+1}|\mathbf{x}_0)}&\left[-\mathbb{E}_{Q_\phi(\mathbf{r}_t|\mathbf{x}_{t+1},\mathbf{x}_0)}\left[D_{KL}\left(Q(\mathbf{x}_t|\mathbf{r}_t,\mathbf{x}_{t+1},\mathbf{x}_0)||P_\theta(\mathbf{x}_t|\mathbf{r}_t,\mathbf{x}_{t+1})\right)\right]\right. \\
&\left.-D_{KL}\left(Q_\phi(\mathbf{r}_t|\mathbf{x}_{t+1},\mathbf{x}_0)||P_\psi(\mathbf{r}_t|\mathbf{x}_{t+1})\right)\right].
\end{aligned}$$

Applying conditional independence over dimensions of $\mathbf{x}_t$ given $\mathbf{r}_t$ gives,

$$\begin{aligned}
L_t = \mathbb{E}_{Q_\phi(\mathbf{x}_{\geq t+1},\mathbf{r}_{\geq t+1}|\mathbf{x}_0)}&\left[-\mathbb{E}_{Q_\phi(\mathbf{r}_t|\mathbf{x}_{t+1},\mathbf{x}_0)}\left[\sum_{n=1}^{N} D_{KL}\left(Q(x_t^n|r_t^n,x_{t+1}^n,x_0^n)||P_\theta(x_t^n|r_t^n,x_{t+1}^n)\right)\right]\right. \\
&\left.-D_{KL}\left(Q_\phi(\mathbf{r}_t|\mathbf{x}_{t+1},\mathbf{x}_0)||P_\psi(\mathbf{r}_t|\mathbf{x}_{t+1})\right)\right].
\end{aligned}$$

For Q and P_θ given by

$$Q(x_t^n|r_t^n, x_{t+1}^n, x_0^n) = \delta_{x_{t+1}^n}(1 - r_t^n) + \delta_{x_0^n} r_t^n,$$
$$P_\theta(x_t^n|r_t^n, \mathbf{x}_{t+1}) = \delta_{x_{t+1}^n}(1 - r_t^n) + \mu_\theta^n(\mathbf{x}_{t+1}, t) r_t^n,$$

the KL divergences are given in closed form by

$$\begin{aligned} D_{KL}\left(Q(x_t^n|r_t^n, x_{t+1}^n, x_0^n) \,||\, P_\theta(x_t^n|r_t^n, \mathbf{x}_{t+1})\right) &= D_{KL}\left(\delta_{x_0^n} \,||\, \mu_\theta^n(\mathbf{x}_{t+1}, t)\right) r_t^n, \\ &= -r_t^n \log \mu_\theta^n(x_{t+1}, t)_{x_0^n}, \end{aligned}$$

subbing this into L_t gives

$$L_t = \mathbb{E}_{Q_\phi(\mathbf{x}_{\geq t+1}, \mathbf{r}_{\geq t+1}|\mathbf{x}_0)}\left[\mathbb{E}_{Q_\phi(\mathbf{r}_t|\mathbf{x}_{t+1}, \mathbf{x}_0)}\left[\sum_{n=1}^N r_t^n \log \mu_\theta^n(\mathbf{x}_{t+1}, t)_{x_0^n}\right]\right. \tag{16}$$
$$\left. - D_{KL}\left(Q_\phi(\mathbf{r}_t|\mathbf{x}_{t+1}, \mathbf{x}_0)||P_\psi(\mathbf{r}_t|\mathbf{x}_{t+1})\right)\right] \tag{17}$$

We assume conditionally independent r_t^n, as this leads to a lower variance final expression for L_t, where expectations with respect to all r_t^n are computed in closed form. The conditional distributions for $\mathbf{r}_t$ are therefore paramaterised as follows,

$$Q_\phi(\mathbf{r}_t|\mathbf{x}_{t+1}, \mathbf{x}_0) = \prod_{n=1}^N Q_\phi(r_t^n|\mathbf{x}_{t+1}, \mathbf{x}_0),$$
$$P_\theta(\mathbf{r}_t|\mathbf{x}_{t+1}) = \prod_{n=1}^N P_\theta(r_t^n|\mathbf{x}_{t+1}),$$

where

$$Q_\phi(r_t^n|\mathbf{x}_{t+1}, \mathbf{x}_0) = \text{Bern}\left(q_\phi^{t,n}(\mathbf{x}_{t+1}, \mathbf{x}_0)\mathbf{1}_{x_{t+1}^n = m}\right),$$
$$P_\theta(r_t^n|\mathbf{x}_{t+1}) = \text{Bern}\left(p_\psi^{t,n}(\mathbf{x}_{t+1})\mathbf{1}_{x_{t+1}^n = m}\right).$$

The probabilities $p_\psi^{t,n}$ and $q_\phi^{t,n}$ are implemented using neural networks. We can now simplify each term in the outer expectation of equation 18,

$$\begin{aligned} \mathbb{E}_{Q_\phi(\mathbf{r}_t|\mathbf{x}_{t+1}, \mathbf{x}_0)}\left[\sum_{n=1}^N r_t^n \log \mu_\theta^n(\mathbf{x}_{t+1}, t)_{x_0^n}\right] &= \sum_{n=1}^N \mathbb{E}_{\prod_{n=1}^N Q_\phi(r_t^n|\mathbf{x}_{t+1}, \mathbf{x}_0)}\left[r_t^n \log \mu_\theta^n(\mathbf{x}_{t+1}, t)_{x_0^n}\right] \\ &= \sum_{\substack{n=1, \\ \text{s.t.} x_{t+1}^n = m}}^N q^{t,n}(\mathbf{x}_{t+1}, \mathbf{x}_0) \log \mu_\theta^n(\mathbf{x}_{t+1})_{x_0^n}. \end{aligned}$$

$$\begin{aligned} D_{KL}\left(Q_\phi(\mathbf{r}_t|\mathbf{x}_{t+1}, \mathbf{x}_0) \,||\, P_\theta(\mathbf{r}_t|\mathbf{x}_{t+1})\right) &= D_{KL}\left(\prod_{n=1}^N Q_\phi(r_t^n|\mathbf{x}_{t+1}, \mathbf{x}_0) \,||\, \prod_{n=1}^N P_\theta(r_t^n|\mathbf{x}_{t+1})\right), \\ &= \sum_{\substack{n=1, \\ \text{s.t.} x_{t+1}^n = m}}^N D_{KL}\left(Q_\phi(r_t^n|\mathbf{x}_{t+1}, \mathbf{x}_0) \,||\, P_\theta(r_t^n|\mathbf{x}_{t+1})\right) \end{aligned}$$

Where in the last line we've used the fact that the KL divergences in the sum are 0 if $x_t^n \neq m$.

Subbing these expressions into equation (13) then gives,

$$L_t = \mathbb{E}_{Q(\mathbf{x}_{\geq t+1}, \mathbf{r}_{\geq t+1}|\mathbf{x}_0)} \Bigg[\sum_{\substack{n=1, \\ \text{s.t.} x_{t+1}^n = m}}^{N} q^{t,n}(\mathbf{x}_{t+1}, \mathbf{x}_0) \log \mu_\theta^n(\mathbf{x}_{t+1})_{x_0^n} \tag{18}$$

$$- D_{KL}\left(Q(r_t^n|\mathbf{x}_{t+1}, \mathbf{x}_0)||P_\psi(r_t^n|\mathbf{x}_{t+1})\right)\Bigg]. \tag{19}$$

The finial total loss function is then

B Masked Diffusion Gradient Estimation

We first introduce some notations for the sake of brevity,

$$F^1(\mathbf{x}_{t+1}, \mathbf{x}_0, \mathbf{q}_\phi^t) = \sum_{\substack{n=1 \\ \text{s.t. } x_{t+1}^n = m}}^{N} q_\phi^{t,n}(\mathbf{x}_{t+1}, \mathbf{x}_0) \log \mu_\theta^n(\mathbf{x}_{t+1})_{x_0^n},$$

$$F^2(\mathbf{x}_{t+1}, \mathbf{q}_\phi^t, \mathbf{p}_\psi^t) = - \sum_{\substack{n=1 \\ \text{s.t. } x_{t+1}^n = m}}^{N} D_{KL}\left(Q_\phi(r_t^n|\mathbf{x}_{t+1}, \mathbf{x}_0)||P_\psi(r_t^n|\mathbf{x}_{t+1})\right),$$

$$F(\mathbf{x}_{t+1}, \mathbf{x}_0, \mathbf{q}_\phi^t, \mathbf{p}_\psi^t) = F^1(\mathbf{x}_{t+1}, \mathbf{x}_0, \mathbf{q}_\phi^t) + F^2(\mathbf{x}_{t+1}, \mathbf{q}_\phi^t, \mathbf{p}_\psi^t),$$

where we suppress dependence of $p_\phi^{t,n}$ and $p_\psi^{t,n}$ on $\mathbf{x}_{t+1}$ and $\mathbf{x}_0$ for notational simplicity. We additionally denote the vector of unmasking probabilities over token indices using bold typeface.

The gradients of the loss are computed as follows,

$$\begin{aligned}
\eta_\psi(\mathbf{x}_0, \psi, \phi) &= \nabla_\psi \mathcal{L}(\mathbf{x}_0, \psi, \phi), \\
&= \sum_{t=1}^{T-1} \mathbb{E}_{Q_\phi(\mathbf{x}_{\geq t+1}|\mathbf{x}_0)} \left[\nabla_\psi F^2(\mathbf{x}_{t+1}, \mathbf{q}_\phi^t, \mathbf{p}_\psi^t)\right], \\
&= (T-1)\mathbb{E}_{\substack{Q_\phi(\mathbf{x}_{\geq t+1}|\mathbf{x}_0) \\ t \sim \text{Unif}(0, T-1)}} \left[\nabla_\psi F^2(\mathbf{x}_{t+1}, \mathbf{q}_\phi^t, \mathbf{p}_\psi^t)\right],
\end{aligned}$$

$$\begin{aligned}
\eta_\phi(\mathbf{x}_0, \psi, \phi) &= \nabla_\phi \mathcal{L}(x_0, \psi, \phi), \\
&= (T-1)\mathbb{E}_{\substack{Q_\phi(\mathbf{x}_{\geq t+1}|\mathbf{x}_0) \\ t \sim \text{Unif}(0, T-1)}} \left[\nabla_\phi F(\mathbf{x}_{t+1}, \mathbf{x}_0, \mathbf{q}_\phi^t, \mathbf{p}_\psi^t)\right], \\
&\quad + (T-1)\mathbb{E}_{\substack{Q_\phi(\mathbf{x}_{\geq t+1}|\mathbf{x}_0) \\ t \sim \text{Unif}(0, T-1)}} \left[F(\mathbf{x}_{t+1}, \mathbf{x}_0, \mathbf{q}_\phi^t, \mathbf{p}_\psi^t) \nabla_\phi \log Q_\phi(\mathbf{x}_{\geq t+1}|\mathbf{x}_0)\right], \\
&= \eta_\phi^1(\mathbf{x}_0, \psi, \phi) + \eta_\phi^2(\mathbf{x}_0, \psi, \phi)
\end{aligned}$$

where we've used the log-derivative trick / REINFORCE to compute η_ϕ. η_ψ and η_ϕ^1 are expectations of gradients, and can be estimated using naive monte carlo.

$$\bar{\eta}^k_\psi(\mathbf{x}_0, \psi, \phi) = \frac{T-1}{k} \sum_{i=1}^{k} \nabla_\psi F^2(\hat{\mathbf{x}}^i_{t+1}, \mathbf{q}^t_\phi, \mathbf{p}^t_\psi),$$
$$t \sim \text{Unif}(0, \dots, T-1),$$
$$\left(\hat{\mathbf{x}}^i_{\geq t+1}, \hat{\mathbf{r}}^i_{\geq t+1}\right) \sim Q_\phi(\mathbf{x}_{\geq t+1}, \mathbf{r}_{\geq t+1} | \mathbf{x}_0), \quad i = 1, ..., k.$$

In η^2_ϕ, the gradient in the expectation is scaled by the sum of F's, which typically leads to excessively high variance in a naive monte carlo estimate. Overcoming this requires variance reduction techniques. For simplicity, we use REINFORCE Leave-One-Out (RLOO) Kool et al. (2019).

$$\bar{\eta}^{2,k}_\phi = \frac{T-1}{k} \sum_{i=1}^{k} \Big[F(\hat{\mathbf{x}}^i_{t+1}, \mathbf{x}_0, \mathbf{q}^t_\phi, \mathbf{p}^t_\psi) - \frac{1}{k-1} \sum_{l=1, l\neq j=i}^{k} F(\hat{\mathbf{x}}^l_{t+1}, \mathbf{x}_0, \mathbf{q}^t_\phi, \mathbf{p}^t_\psi) \Big] \nabla_\phi \log Q_\phi(\hat{\mathbf{x}}^i_{\geq t+1} | \mathbf{x}_0)$$
$$t \sim \text{Unif}(0, \dots, T-1),$$
$$\left(\hat{\mathbf{x}}^i_{\geq t+1}, \hat{\mathbf{r}}^i_{\geq t+1}\right) \sim Q_\phi(\mathbf{x}_{\geq t+1}, \mathbf{r}_{\geq t+1} | \mathbf{x}_0), \quad i = 1, ..., k.$$