ADAPTIVE ORDER POLICIES FOR MASKED DIFFUSION

Mohsin Hasan[1,2*], **Jama Hussein Mohamud**[1,2*], **Mirco Ravanelli**[2,3], **Yoshua Bengio**[1,2,4]

[1]Université de Montréal, [2]Mila, [3]Concordia University, [4]LawZero

ABSTRACT

Masked diffusion models have seen great success in capturing data distributions over discrete sequences in domains such as text and proteins. These models generate data by iteratively unmasking tokens starting from a fully masked sequence, with the unmasking order typically chosen at random or using a heuristic based on denoiser probabilities. In this work, we propose a scheme for learning the unmasking order using an additional lightweight policy network on top of a diffusion model. Our proposed loss reweights terms in the masked diffusion loss according to policy probabilities, and results in a policy that prefers positions where the denoiser is more likely to be correct. We study this loss in two settings: (i) training solely the policy while using a frozen pre-trained denoiser, and (ii) training the policy and denoiser jointly with the weighted loss to allow for mutual adaptation. We demonstrate that our approach outperforms common heuristics on problems that are sensitive to token ordering, such as Sudoku and Boolean satisfiability (3-SAT).

1 INTRODUCTION

Diffusion models have established themselves as a powerful paradigm for generative modeling, achieving remarkable success in continuous domains such as images (Ho et al., 2020; Saharia et al., 2022; Rombach et al., 2022) and molecular structures (Watson et al., 2023; Abramson et al., 2024). More recently, *discrete* diffusion models – which operate directly on token sequences by iteratively masking and unmasking – have shown strong results in language modeling (Sahoo et al., 2024; Nie et al., 2025; Shi et al., 2024), protein design (Alamdari et al., 2023; Wang et al., 2024), and drug discovery (Lee et al., 2025).

A key design choice in masked diffusion models (MDM) is the *order* in which tokens are unmasked during generation. The standard approach selects positions uniformly at random. However, practitioners have found that heuristic ordering strategies – such as unmasking the most confident position first (Nie et al., 2025) or the position with the largest probability margin (Kim et al., 2025) – can dramatically improve sample quality on downstream tasks. This effect is particularly pronounced on constraint satisfaction problems such as Sudoku and Boolean satisfiability (3-SAT), where the unmasking order directly impacts whether the model can propagate constraints correctly.

Despite the empirical success of heuristic orderings, these remain hand-designed and may be suboptimal for a given model and dataset. A natural question arises: *can we learn the unmasking order?* That is, rather than relying on fixed heuristics, can we train a lightweight auxiliary network to predict which positions to unmask, conditioned on the current partially masked sequence?

In this work, we propose a simple approach for learning adaptive unmasking orderings in MDMs. Our approach introduces a policy network $q^{\phi}(i \mid x_t)$ that is trained on top of a masked diffusion model via a modified cross-entropy objective that weights policy probabilities by the cross entropy of the denoiser at each token position. We investigate the application of this objective for fine-tuning a policy layer on top of a pretrained denoiser. We demonstrate that on the tasks of Sudoku and 3-SAT, our trained policy outperforms existing heuristics, while adding very few parameters ($< 1\%$ of the MDM's total parameters) and requiring very few training iterations (converging within a few hundred training iterations, compared to the hundreds of thousands required for MDM training). We additionally propose a modified objective which allows for training the denoiser such that it is aware

[*]Equal contribution.
Correspondence to {mohsin.hasan, hussein-mohamu.jama}@mila.quebec

of the policy. With this objective, we demonstrate that joint training further improves the model's performance.

2 METHOD

2.1 MASKED DIFFUSION MODELS

Throughout this work, we denote a sequence of length L as $x = (x^1, \ldots, x^L) \in \mathcal{V}^L$, with tokens taking values in some vocabulary set $x^i \in \mathcal{V}$. We consider the case of masked diffusion, where a special masking token m is included in the vocabulary set. Other notation includes: the Kronecker symbol $\delta(i, j)$ (equal to 1 for $i = j$ and 0 otherwise), $\mathrm{Cat}(x; p)$ to denote the categorical distribution with probabilities p, and Δ^k to denote the probability simplex over k dimensions.

Masked diffusion models (MDMs) use a noising process to map the data distribution $p_{\mathrm{data}}(x)$ at time 0 to the delta distribution at the fully masked state $M = (m, \ldots, m)$, $p_1(x) = \delta(x, M)$ at time 1. A typical noising process consists of converting a data token x_0^i into the masked token with some probability $1 - \alpha_t$, independently over dimensions (Sahoo et al., 2024): $p(x_t \mid x_0) = \prod_{i=1}^{L} \alpha_t \delta(x_t^i, x_0^i) + (1 - \alpha_t)\delta(x_t^i, m)$. The parameter α_t denotes a decreasing noise schedule, with $\alpha_0 = 1$ and $\alpha_1 = 0$. A typical choice is the linear schedule $\alpha_t = 1 - t$.

For reversing this process, a neural network parameterizes a distribution over clean data x_0 conditioned on the partially masked sequence x_t. In particular, the network outputs an independent distribution over each token position i, as $\mu^\theta(x_t)[i, \cdot] \in \Delta^{|\mathcal{V}|}$, which satisfies $\mu^\theta(x_t)[i, m] = 0$ (the clean data cannot contain masks) and $\mu^\theta(x_t)[i, x_t^i] = 1$ if $x_t^i \neq m$ (the clean data approximation retains unmasked positions in x_t). The function μ^θ is referred to as the denoiser.

Given a denoiser, the reverse transition over two nearby time-steps $s < t$ is (Sahoo et al., 2024):

$$p^\theta(x_s^i \mid x_t) = \begin{cases} \mathrm{Cat}\left(x_s^i; \frac{1-\alpha_s}{1-\alpha_t}\delta(\cdot, m) + \frac{\alpha_s - \alpha_t}{1-\alpha_t}\mu^\theta(x_t)[i, \cdot]\right) & \text{if } x_t^i = m \\ \mathrm{Cat}(x_s^i; \delta(\cdot, x_t^i)) & \text{if } x_t^i \neq m \end{cases} \tag{1}$$

Let $\mathrm{CE}(i, p)$ denote the cross entropy loss with sample i and probability p: $\mathrm{CE}(i, p) = -\log p[i]$. The goal is to train $\mu^\theta(x_t)$ to match the factored posterior $\prod_{i=1}^{L} p(x_0^i \mid x_t)$. For such a denoiser, the transitions in Equation (1) correctly reverse the noising process and recover the data distribution at time 0 (Sahoo et al., 2024; Shi et al., 2024). The training objective for μ^θ is the weighted cross entropy loss:

$$\mathcal{L}_{\mathrm{MDM}}(\theta) = \mathbb{E}_{t \sim \mathcal{U}[0,1], x_t \sim p(x_t|x_0), x_0 \sim p_{\mathrm{data}}} \left[\frac{-\alpha_t'}{1 - \alpha_t} \sum_{i=1}^{L} \delta(x_t^i, m) \mathrm{CE}(x_0^i, \mu^\theta(x_t)[i, \cdot]) \right] \tag{2}$$

With a trained denoiser, the generation process consists of starting with a completely masked sequence $x_1 = M$ and iteratively unmasking tokens in the sequence over T steps to obtain a final sample x_0. One step of this sampling procedure is done by simulating Equation (1), which involves:

(i) Sampling an approximation of clean data from the denoiser $\hat{x}_0 \sim \mu^\theta(x_t)$

(ii) Randomly choosing which (currently masked) positions I in x_t to unmask (by replacing $x_t^i = m$ with $\hat{x}_0^i$ for all $i \in I$).

Properly simulating Equation (1) requires random selection of the set of unmasking positions I (Sahoo et al., 2024). However, a number of works have found success in selecting I through some heuristic informed by the denoiser logits $\mu^\theta(x_t)$ (Nie et al., 2025; Kim et al., 2025; Ben-Hamu et al., 2025). These involve calculating a score s_i and then prioritizing unmasking positions i with higher score (possibly with added noise).

Some options for the score calculation which have been investigated in previous work include choices such as (i) **Top probability**: The probability of the sampled clean tokens $s_i = \mu^\theta(x_t)[i, \hat{x}_0^i]$ (Nie et al., 2025), (ii) **Top probability margin**: The probability margin, i.e. the gap between the highest probability and the second highest probability $s_i = \mu^\theta(x_t)[i, j_1] - \mu^\theta(x_t)[i, j_2]$ where $j_1 = \arg\max_j \mu^\theta(x_t)[i, j]$ and $j_2 = \arg\max_{j \neq j_1} \mu^\theta(x_t)[i, j]$ (Kim et al., 2025), and (iii) **Entropy**: The negative entropy of the position $s_i = -H(\mu^\theta(x_t)[i, \cdot])$ (Ben-Hamu et al., 2025).

These heuristics have been shown to yield better performance on a number of downstream tasks (such as coding and math), or on tasks such as Sudoku, where generating a valid solution is highly dependent on the order of unmasking (Nie et al., 2025; Kim et al., 2025).

The problem of selecting the unmasking ordering also informs the efficiency of performing inference, since being able to unmask multiple tokens leads to fewer calls to the denoising model to generate a complete sequence. Intuitively, we expect certain token positions to be independent of others, and we would expect unmasking them in parallel to retain the same performance as unmasking one at a time.

2.2 Learnable Adaptive Order Policies

Given the importance of the unmasking ordering, we ask the following question: armed with a dataset, and a pre-trained denoiser μ^θ, can we train a lightweight *policy network* $q^\phi(i \mid x_t)$ which outputs a distribution over the next token position i to unmask for x_t? The hope is to add a small number of additional trainable parameters, and spend some additional training iterations to obtain better performance on challenging tasks.

We can note that MDMs are sensitive to the token ordering precisely due to imperfections in the denoiser model (otherwise they would be able to generate perfectly through uniformly sampled positions). Therefore, a reasonable objective for the policy is one that accounts for the loss of the denoiser model. Based on this, we propose to train the policy to place higher probability on positions which obtain a smaller loss, as measured by the cross entropy. This is captured in the objective:

$$\mathcal{L}_{\text{ORDER}}(\phi) = \mathbb{E}_{t\sim\mathcal{U}[0,1],x_t\sim p(x_t|x_0),x_0\sim p_{\text{data}}}\left[\frac{-\alpha_t'}{1-\alpha_t}\sum_{i=1}^{L} q^\phi(i \mid x_t)\text{CE}(x_0^i, \mu^\theta(x_t)[i,\cdot])\right] \tag{3}$$

This is identical to the MDM objective Equation (2) except for the fact that the sum over masked positions is weighted by the policy q^ϕ. In particular, for a uniform policy over masked tokens, we recover, up to multiplication, the vanilla MDM loss.

For a frozen denoiser network θ, the policy network q^ϕ is trained to predict where the current denoiser is most likely to make errors. The optimal policy places all probability on the position with smallest cross entropy loss. We will refer to this theoretically optimal policy as the **oracle**: $q^{\text{oracle}}(i \mid x_t) = \delta(i, \arg\min_{j,x_t^j=m} \text{CE}(x_0^j, \mu^\theta(x_t)[j,\cdot]))$. We empirically validate the choice of this loss by evaluating the oracle policy (with access to ground truth data x_0), and confirming that it improves metrics relative to other heuristic samplers in Section 4. Other approaches for training unmasking policies typically rely on more expensive gradient estimation, or RL procedures for tasks involving a reward. These are discussed in Section 3.

We can observe that, for training denoiser parameters θ, the objective in Equation (3) has the effect of up-weighting the positions likely to be unmasked by the policy q^ϕ, by assigning a larger penalty to errors at these positions. This suggests a natural extension in which the policy does not only learn from denoiser errors, but also shapes which errors the denoiser itself prioritizes during training.

Finally we note that for sampling multiple positions, we can use the policy probabilities as score values $s_i = q^\phi(i \mid x_t)$ and use them in a similar way to other heuristics (for instance, unmasking the k positions with the largest policy probabilities).

2.3 Policy-Aware Denoiser Training

The policy objective in Equation (3) answers where the model should unmask next, but it also raises two related questions: can the policy improve the denoiser itself, and can the denoiser be trained while explicitly knowing that a policy will be used at sampling time? We study this by a minimal modification of the denoiser loss.

Instead of optimizing the denoiser with the vanilla MDM objective, we use a policy-weighted objective,

$$\mathcal{L}_{\text{PA-MDM}}(\theta) = \mathbb{E}\left[\frac{-\alpha_t'}{1-\alpha_t}\sum_{i=1}^{L}\delta(x_t^i, m)\left(1 + q^\phi(i \mid x_t)\right)\text{CE}(x_0^i, \mu^\theta(x_t)[i,\cdot])\right]. \tag{4}$$

The additional factor $1 + q^\phi(i \mid x_t)$ leaves the original denoising loss in place, while increasing the contribution of positions that the policy considers important for future unmasking decisions. In implementation, the policy probabilities are detached inside the denoiser loss, so gradients with respect to θ do not backpropagate through q^ϕ.

The policy itself is still trained with Equation (3), with the denoiser losses detached when optimizing ϕ. Joint training therefore lets the policy identify consequential positions, while simultaneously teaching the denoiser to allocate more capacity to them.

To separate the effect of *learned* policy guidance from the effect of simply reweighting the denoiser loss, we can also compare against heuristic-aware denoiser objectives, similar to those in (Peng et al., 2025). Concretely, we can replace $q^\phi(i \mid x_t)$ in Equation (4) by normalized weights derived from standard confidence scores computed from the denoiser logits, such as the top log-probability or the top-two margin. This yields a matched control where the denoiser is still trained with emphasis on selected positions, but the emphasis is determined by a fixed heuristic rather than a learned policy.

Such a comparison isolates the main question of interest: whether informing the denoiser about the existence of a learned unmasking policy during training provides benefits beyond the gains obtainable from generic confidence-based reweighting alone.

3 Related Works

Wang et al. (2025) also train a policy for the unmasking order, by treating the order as a latent variable z. They propose a variational method for optimizing it, which requires parameterizing the posterior approximation $q^\phi(z \mid x)$, in addition to the trainable policy over orders $p^\theta(z \mid x)$. The former is only used as part of the training objective for the latter, and not during inference. In addition, the optimization of the variational posterior requires gradient estimation techniques to reduce variance, and complicates the optimization loop. Our loss by contrast is much simpler, though it doesn't optimize a theoretical ELBO objective.

Another set of works assume access to verifiable reward functions rather than a dataset (Hong et al., 2025; Jazbec et al., 2025). These frame the generation process of a masked diffusion model as a Markov Decision Process, and optimize the unmasking policy using RL objectives. Our work focuses on the setting where data is available, since the aim is to expand to modalities where an explicit reward function is not as obvious, such as proteins.

Peng et al. (2025) propose a modification of the MDM loss which accounts for using heuristics in determining the unmasking order (rather than random unmasking). The loss resembles our objective Equation (4), with the main difference that we focus on combining this loss with a jointly trained policy model. Further investigation of this joint training objective forms a direction for future work.

4 Experiments

We evaluate our adaptive ordering approach on two constraint satisfaction tasks known to be sensitive to token ordering: Sudoku puzzle solving and 3-SAT (Boolean satisfiability with 3 literals per clause) (Kim et al., 2025; Ye et al., 2024). We use a 6M-parameter GPT-2 denoiser trained as an MDM with $T{=}20$ steps; full dataset descriptions and training details are provided in Appendix A.

Policy architecture. We parameterize q^ϕ as a lightweight per-token MLP that conditions on both the denoiser's confidence scores (max log-probability) and the hidden states from the last layer of the base model. Specifically, each scalar confidence score is projected to the hidden dimension ($d{=}384$) via a linear layer, summed with the corresponding hidden-state vector, and passed through a two-layer MLP (hidden dimension 128, ReLU activation) that outputs a per-position routing logit. The policy adds $\sim$50K parameters ($<$1% of the base model) and is trained on top of the frozen denoiser using Equation (3). We also experimented with a transformer-based policy variant (Jazbec et al., 2025), which did not yield further improvement (see Appendix B).

Baselines and decoding. At inference, we compare against the **Top probability**, **Top prob. margin**, and **Oracle** ordering strategies described in Section 2. Unless otherwise noted, all methods use deterministic top-k decoding with a linear schedule over $T{=}20$ reverse steps. For our joint training

Table 1: Results for policy-only training on Sudoku and 3-SAT, compared to other decoding strategies under deterministic and stochastic (Gumbel noise, scale 0.5) variants. All use a linear schedule with T=20 steps. Best non-oracle result per column in **bold**.

	Sudoku		3-SAT	
Ordering Strategy	**Det.**	**Stoch.**	**Det.**	**Stoch.**
Top probability	89.84%	18.26%	75.9%	72.8%
Top prob. margin	88.67%	88.38%	76.0%	75.6%
Learned policy (ours)*	**90.82%**	**90.53%**	**76.1%**	**75.9%**
Oracle policy (ours)	100.0%	N/A	82.3%	N/A

*Policy adds <50K params (<1% of base model).

Table 2: Results for policy-aware denoiser training. "Baseline" denotes the original MDM objective with no scaling. "High conf." and "Margin" replace the policy weights in Equation (4) with normalized heuristic weights derived from max log-probability and top-two margin, respectively. "Policy" uses the learned policy weights. We report accuracy under the matched decoding rule for each method. Best result per column in **bold**.

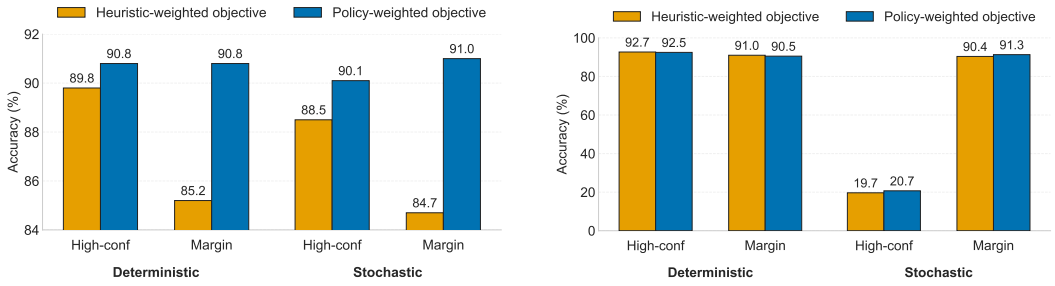
	Sudoku		3-SAT	
Training Objective	**Det.**	**Stoch.**	**Det.**	**Stoch.**
Baseline (no scaling)	92.72	18.35	88.8	87.8
High conf. scaling	92.68	19.67	89.8	88.5
Margin scaling	91.00	90.38	85.2	84.7
Policy-aware scaling (ours)	**92.87**	**93.36**	**90.9**	**90.9**

experiments, **baseline** refers to the original MDM loss with no additional scaling, while the heuristic and policy variants use the modified denoiser objective from Equation (4).

4.1 RESULTS

Policy-only training. Table 1 presents our results for training a policy on a pre-trained denoiser model, on both tasks using Equation (3). On Sudoku, our learned ordering policy achieves the highest non-oracle accuracy at **90.82%**, outperforming both the top-probability heuristic (89.84%) and the top-margin heuristic (88.67%). On 3-SAT, the learned policy (76.1%) again outperforms top-probability (75.9%) and matches or exceeds the margin heuristic (76.0%). The oracle policy achieves 100% on Sudoku and 82.3% on 3-SAT, indicating substantial room for further improvement in learning the optimal unmasking order on both tasks.

On the role of decoding strategy. A striking observation in Table 1 is the discrepancy between our top-probability result on Sudoku (89.84%) and the 18.51% reported by Kim et al. (2025). We find that this gap is primarily an artifact of the *decoding strategy*, not an inherent limitation of the heuristic. As shown in Table 1, when we switch from deterministic to stochastic decoding (adding Gumbel noise with scale 0.5), the top-probability heuristic drops from 89.84% to 18.26% on Sudoku – closely matching the 18.51% of Kim et al. (2025), who use stochastic decoding with the same noise coefficient. The margin heuristic, by contrast, is robust to this noise (88.67% $\rightarrow$ 88.38%). This reveals that the reported large advantage of the margin heuristic over top-probability is largely attributable to the latter's sensitivity to stochastic perturbations in decoding, rather than a fundamentally superior ordering strategy. Under deterministic decoding, both heuristics perform comparably, with top-probability slightly ahead. This pattern is less extreme on 3-SAT, where top-probability degrades modestly under stochastic decoding (75.9% $\rightarrow$ 72.8%), while the margin heuristic remains stable (76.0% $\rightarrow$ 75.6%). Notably, our learned policy is robust to stochastic noise on both tasks – Sudoku: 90.82% $\rightarrow$ 90.53%; 3-SAT: 76.1% $\rightarrow$ 75.9% – retaining strong performance in both decoding regimes while consistently outperforming both heuristics.



(a) 3-SAT (b) Sudoku

Figure 1: Heuristic transfer under policy-aware training. We compare heuristic-specific training objectives against our policy-weighted objective when decoding with the same heuristic. Policy-aware training improves both high-confidence and margin decoding across tasks, especially in stochastic decoding regimes.

Policy-aware denoiser training. Table 2 evaluates the loss modification from Equation (4). The main pattern is that learned policy reweighting improves the denoiser more reliably than heuristic reweighting. On Sudoku, policy-aware scaling yields the best accuracy in both decoding regimes, improving from 92.72% to **92.87%** under deterministic decoding and, more importantly, from 18.35% to **93.36%** under stochastic decoding. Heuristic scaling does not recover this behavior: high-confidence scaling remains brittle under stochastic decoding (19.67%), whereas margin scaling is robust (90.38%) but still falls short of the learned policy. On 3-SAT, the pattern is more modest but consistent: policy-aware scaling reaches **90.9%** accuracy in both decoding regimes, outperforming both the unscaled baseline and the heuristic-weighted controls.

An important feature of this result is that policy-aware training does not only help when decoding with the learned policy. As shown in Figures 1a and 1b, the denoiser trained with policy-aware scaling also improves heuristic decoding, most clearly on 3-SAT and in the stochastic Sudoku setting, while remaining competitive in the deterministic Sudoku setting. This suggests that the policy-weighted objective is not merely matching one decoder to one sampler, but can improve the underlying denoiser in a way that transfers across different decoding heuristics.

Efficiency as a function of diffusion steps. We also study how ordering quality interacts with the number of reverse steps T, which directly controls inference cost. As shown in Figure 2, learned ordering improves efficiency by achieving higher accuracy for the same step budget. On Sudoku, the learned policy substantially outperforms both heuristic baselines and nearly matches the oracle at T=100. On 3-SAT, the learned policy consistently improves over the low-confidence heuristic and is competitive with, or slightly better than, the margin heuristic across moderate and large step budgets. In both tasks, the oracle remains better than the learned policy, indicating that there is still substantial room to improve learned unmasking strategies. Overall, these results reinforce that ordering is not only an accuracy issue, but also an efficiency issue: stronger policies can achieve better performance with fewer denoising steps.

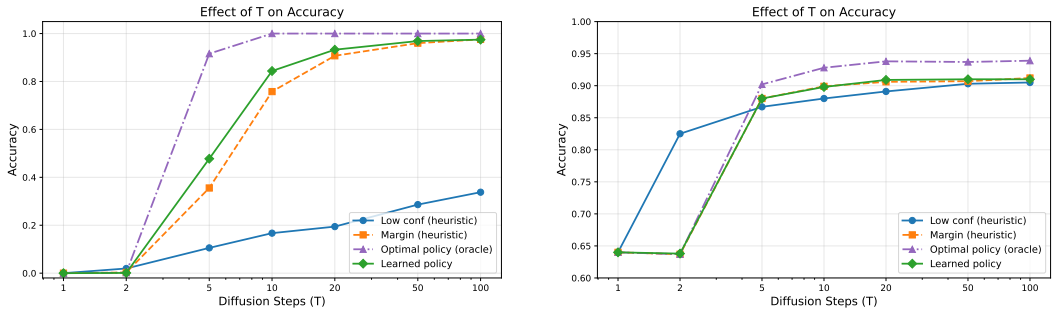


Figure 2: Accuracy as a function of the number of reverse diffusion steps T on Sudoku (left) and 3-SAT (right). Better ordering is especially valuable at small step budgets, where improved unmasking policies can recover substantially more accuracy for the same inference cost.

Discussion. Several observations emerge from these results. First, the learned policy consistently improves over all heuristics on both tasks, achieved with a lightweight auxiliary network adding less than 1% parameters, demonstrating that the unmasking order can be improved by learning. Second, the gap between the learned policy and the oracle – particularly large on Sudoku (90.82% vs. 100%) and still substantial on 3-SAT (76.1% vs. 82.3%) – suggests that significantly better policies are achievable, motivating future work on more expressive policy architectures and training procedures. Third, our analysis on decoding strategies highlights the importance of carefully controlling for inference-time design choices when comparing ordering heuristics – a point that has been underappreciated in prior work. Fourth, the policy-aware denoiser training ablation indicates that the policy is useful not only at inference time, but also as a training signal for the denoiser itself; the learned reweighting consistently outperforms matched heuristic-based controls. Finally, the fact that our approach uses only $T{=}20$ diffusion steps (compared to 50 in Kim et al. (2025)) while achieving competitive or superior accuracy suggests that learned orderings may also enable more efficient inference. In all cases, the policy converged within a few hundred iterations, requiring only a small fraction compared to the base model's training budget.

5 Conclusion

This work studies a method for training a policy over token orderings for masked diffusion models, by learning to sample positions with lower cross-entropy loss. We demonstrate that our approach outperforms common heuristics for logical tasks which are sensitive to the unmasking ordering, at the cost of a few training iterations for a lightweight auxiliary network. For future work, we intend to expand the evaluation to more difficult tasks and data modalities, as well as investigate theoretical properties of the joint training.

Acknowledgements

The research was enabled in part by computational resources provided by the Digital Research Alliance of Canada (`https://alliancecan.ca`) and Mila (`https://mila.quebec`).

REFERENCES

Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J Ballard, Joshua Bambrick, et al. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature*, 630(8016):493–500, 2024.

Sarah Alamdari, Nitya Thakkar, Rianne van den Berg, Neil Tenenholtz, Bob Strome, Alan Moses, Alex Xijie Lu, Nicolo Fusi, Ava Pardis Amini, and Kevin K Yang. Protein generation with evolutionary diffusion: sequence is all you need. *BioRxiv*, pp. 2023–09, 2023.

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. URL https://arxiv.org/abs/2505.24857.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arXiv:2006.11239*, 2020.

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. URL https://arxiv.org/abs/2510.05725.

Metod Jazbec, Theo X. Olausson, Louis Béthune, Pierre Ablin, Michael Kirchhof, João Monteiro, Victor Turrisi, Jason Ramapuram, and Marco Cuturi. Learning unmasking policies for diffusion language models. *arXiv preprint arXiv:2512.09106*, 2025. URL https://arxiv.org/abs/2512.09106.

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*, 2025. URL https://arxiv.org/abs/2502.06768.

Seul Lee, Karsten Kreis, Srimukh Prasad Veccham, Meng Liu, Danny Reidenbach, Yuxing Peng, Saee Paliwal, Weili Nie, and Arash Vahdat. Genmol: A drug discovery generalist with discrete diffusion. *arXiv preprint arXiv:2501.06158*, 2025.

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. URL https://arxiv.org/abs/2502.09992.

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, 2025. URL https://arxiv.org/abs/2509.23405.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:36479–36494, 2022.

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

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

Xinyou Wang, Zaixiang Zheng, Fei Ye, Dongyu Xue, Shujian Huang, and Quanquan Gu. Diffusion language models are versatile protein learners. *International Conference on Machine Learning (ICML)*, 2024.

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. URL https://arxiv.org/abs/2503.05979.

Joseph L Watson, David Juergens, Nathaniel R Bennett, Brian L Trippe, Jason Yim, Helen E Eisenach, Woody Ahern, Andrew J Borst, Robert J Ragotte, Lukas F Milles, et al. De novo design of protein structure and function with rfdiffusion. *Nature*, 620(7976):1089–1100, 2023.

Jiacheng Ye, Jiahui Gao, Shansan Gong, Lin Zheng, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Beyond autoregression: Discrete diffusion for complex reasoning and planning. *arXiv preprint arXiv:2410.14157*, 2024.

A Experimental Details

Tasks and datasets. We consider two tasks. **Sudoku**: 9×9 puzzles where the model fills in blank cells given a partially completed grid, represented as a flat sequence of 81 tokens (digits 1–9). **3-SAT**: Boolean satisfiability instances with 9 variables and 3 literals per clause, where the model must find a satisfying assignment given the clause structure. For Sudoku, we follow the dataset and evaluation protocol of Kim et al. (2025); for 3-SAT, we use the dataset of Ye et al. (2024). We report instance-level accuracy: an instance is correct only if the entire solution is valid.

Base model and training. We use a 6M-parameter GPT-2 architecture (3 layers, 384 hidden dimensions, 12 attention heads, vocabulary size 31) as the denoiser $\hat{p}^{\theta}$, trained as a masked diffusion model (MDM) with $T{=}20$ diffusion steps. Training uses focal-loss-style token reweighting ($\alpha{=}0.25$, $\gamma{=}1$) and linear time reweighting, with a learning rate of 10^{-3}, batch size of 1024, cosine learning rate schedule, and mixed-precision (fp16) on a single A100 GPU. For Sudoku, the denoiser is trained first for 115K steps, after which the policy is trained on top of the frozen denoiser for 24K steps using Equation (3). For 3-SAT, the denoiser is trained for 58.5K steps and the policy for 7.5K steps. Neither stage was trained to full convergence on either task.

B Transformer Policy Variant

We also evaluated a transformer-based policy architecture as an alternative to the per-token MLP described in the main text. This variant replaces the MLP with a single transformer encoder layer, allowing the policy to attend across positions when making ordering decisions. We tested two configurations:

Table 3: Comparison of policy architectures on Sudoku (deterministic-linear decoding, $T{=}20$ steps).

Policy Architecture	Accuracy
Per-token MLP (scores + hidden)	**90.82%**
Score Transformer (scores only)	89.84%
Score + Hidden Transformer (scores + hidden)	90.23%

- **Score Transformer**: Takes only per-token confidence scores as input, projecting each scalar to $d{=}128$ before a transformer encoder layer.
- **Score + Hidden Transformer**: Additionally conditions on the denoiser's hidden states ($d{=}384$), summing projected scores with hidden representations before the transformer layer.

Results are shown in Table 3. The Score + Hidden Transformer achieves 90.23%, which is slightly lower than the simpler per-token MLP (90.82%). This suggests that cross-position attention in the policy does not provide additional benefit for this task, and the per-token hidden-state representation already captures sufficient information for effective ordering decisions.