Improved Sampling from Masked Diffusion Models with Position Contrastive Guidance

Dhruvesh Patel[*1] **Tahira Naseem**[2] **Gaurav Pandey**[2] **Md Arafat Sultan**[2]
Andrew McCallum[1] **Ramón Fernandez Astudillo**[2]

[1] University of Massachusetts Amherst
[2] IBM Research AI

Abstract

Masked Diffusion Models (MDMs), which generate multiple tokens at a time, hold the promise of accelerating text generation. However, the performance of MDMs is sensitive to the order in which the tokens are generated. We observe that the MDMs are overconfident about the masked positions on the extreme ends of the output sequence. MDMs also express uncertainty by producing similar probability scores for tokens regardless of the query position. Utilizing these insights, we propose Position Contrastive Guidance (PCG), which has two components, a soft order bias that favors left-to-right decoding, and a classifier free guidance that renormalizes the probabilities using position uncertainty to generate more informative tokens earlier in the generation. PCG can be easily plugged into any existing uncertainty-guided sampling strategy. Our experiments on GSM8k, MATH500, and HumanEval show that PCG improves both accuracy as well as throughput for the base as well as instruct version of DREAM-7B and LLaDA-8B models. We also present ablations to identify the contribution of each of proposed components.

1 Introduction

Masked diffusion models (MDMs) [1, 2] have shown performance on par with autoregressive models (ARMs). One key advantage of MDMs is their ability to generate multiple tokens in a single forward pass. However, eliciting accurate predictions while maintaining the throughput requires careful design of the sampling strategy. A typical approach for sampling multiple tokens per step is to use the so called "sort and select" approach, wherein the masked positions are first sorted according to a heuristic ordering function, and then a subset of positions is selected for unmasking using a selection criterion [3, 4]. Both the sort as well as the select steps utilize the model's own uncertainty estimate.

We analyze the sampling trajectories of two MDMs, LLaDA-8B [1] and DREAM-7B [2], and make two observations. First, we observe that the model is overconfident about the masked positions on the extreme left and right ends of the output sequence. Second, the model expresses uncertainty by producing position agnostic probabilities for tokens, i.e., a token gets similar probabilities regardless of which masked position one queries. Based on these observations, we propose Position Contrastive Guidance (PCG), a position reweighting method that that impacts both the sorting as well as the selection step. PCG has two components: (1) a soft left-to-right bias (SLR) that modifies the sorting heuristic to suppress overconfident positions on the extreme right of the masked sequence, and (2) classifier-free guidance utilizing position uncertainty, which discounts the probabilities of tokens that have high probability regardless of the query position.

[*] Work done while an intern at IBM Research.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Structured Probabilistic Inference & Generative Modeling.

Through experiments on GSM8K, MATH500, and HumanEval, using LLaDA-8B and DREAM-7B, we show that PCG improves both the accuracy as well as the throughput of the state-of-the-art "sort and select" style sampling algorithm [3]. We also analyze the contribution of each of the components of PCG to the overall performance and discuss how it varies with the task and the model.

2 Related Work

The performance of MDMs is quite sensitive to the generation order [5], which has resulted in a rich line of work aiming to improve the sampling from MDMs. There have been several recent works that propose utilizing model's own uncertainty estimate to guide the selection of positions to unmask. Kim et al. [5], Zheng et al. [6] experiment with greedy position selection using top probability and top probability margin, respectively, focusing mainly on improving the accuracy. There are works that aim to improve the throughput of MDMs, again by utilizing uncertainty estimates from the model. Ben-Hamu et al. [3] proposes a "sort and select" style sampling algorithm that dynamically decides the number of positions to unmask at each step based on a threshold. The positions are first sorted using the top-probability score, and then a subset of positions is selected until an error budget is exceeded. Wu et al. [4] also proposes a similar approach but using a different error metric, and further augment it with the use of KV-cache. Our work, which focuses on using position uncertainty as a guidance signal can be seen as complementary to these works and can be used in conjunction with them to improve the throughput and accuracy even further.

3 Masked Diffusion Models

Notation The vocabulary, denoted as $\mathbb{V}$, represents the set of all possible tokens except for a special mask token, which is written as v_{mask}. A sequence of tokens of length n is denoted as $\boldsymbol{x} \in \mathbb{V}^n$. A bit vector $\boldsymbol{b} \in \{0,1\}^n$ is used to indicate which positions in a sequence are unknown (masked); specifically, if the i-th entry of $\boldsymbol{b}$ is 0, then the corresponding token in the sequence is masked. A *Masked sequence*, with masked positions specified by zeros in $\boldsymbol{b}$ is denoted as $\boldsymbol{x_b}$. When needed we will use $\boldsymbol{z}$ to denote a masked sequence, without explicitly specifying $\boldsymbol{b}$ and $\boldsymbol{x}$. The notation $\bar{\boldsymbol{b}}$ refers to the bitwise complement of $\boldsymbol{b}$, flipping all bits. $\mathbb{I}_{\boldsymbol{b}} \subseteq [\![n]\!]$ (resp. $\mathbb{I}_{\boldsymbol{z}}$) is the set of indices of the positions where $\boldsymbol{b}$ is 1 (resp. $\boldsymbol{z}$ is not v_{mask}). Similarly we will use $\mathbb{O}_{\boldsymbol{b}} = [\![n]\!] \setminus \mathbb{I}_{\boldsymbol{b}}$ (resp. $\mathbb{O}_{\boldsymbol{z}} = [\![n]\!] \setminus \mathbb{I}_{\boldsymbol{z}}$) to denote the set of indices of the positions where $\boldsymbol{b}$ is 0 (resp. $\boldsymbol{z}$ is v_{mask}). A table of notations can be found in Appendix A.1 for quick reference.

3.1 Time Agnostic Masked Diffusion Models

Masked diffusion models can be categorized into two broad categories. The first category, which inspired the use of the word *diffusion* in the name includes the models wherein the generative process is formulated as the reversal of a stochastic (noising) process that independently converts any token into the *mask* token. There are sub-variants in this category depending on the use of discrete time [7, 8] *vs* continuous time [9–11], or probability parameterization [10, 12] *vs* marginal ratio parameterization [11]. The defining characteristic of this class of models is that the generative network takes in a time parameter as input. Follow-up works [13, 14] show that the time parameter can be seen as a proxy for the number of mask tokens left to fill, and therefore can be removed from the input without losing performance, which brings us to the second class of diffusion models, *Time Agnostic Masked Diffusion Models* (TAMDMs). Removing the time dependence decouples the inference process from the noising process opening up a whole design space for improving inference accuracy as well as speed. In this work, we will focus solely on the TAMDMs (from hereon simply called MDMs). Next we provide a brief overview of training and inference for MDMs.

3.1.1 Training

For $i \in \mathbb{O}_{\boldsymbol{b}}$, let $p_\theta(v \mid \boldsymbol{x}_b, i)$ denote the marginal probability given by the model to token $v \in \mathbb{V}$ at position i given the masked sequence $\boldsymbol{x}_b$. The training objective is to minimize

$$\mathcal{L}(\theta) = -\mathop{\mathbb{E}}_{\boldsymbol{x}\sim p_{\text{data}}} \mathop{\mathbb{E}}_{\boldsymbol{b}\sim\mathcal{B}} \sum_{i\in\mathbb{O}_{\boldsymbol{b}}} w(\boldsymbol{b}) \log p_\theta(x_i \mid \boldsymbol{x}_b, i), \tag{1}$$

where p_{data} is the data distribution on the space $\bar{\mathbb{V}}^n$ and $w(\boldsymbol{b})$ is a weight assigned to the sampled mask pattern $\boldsymbol{b}$ and it depends on the mask generating distribution $\mathcal{B}$.

3.1.2 Inference

For time agnostic MDMs, the problem of inference is simple; the goal is to sample from the conditional joint distribution $P(\boldsymbol{X} = \boldsymbol{x} \mid \boldsymbol{Z} = \boldsymbol{x_b})$, where the conditioning event is the partially masked sequence $\boldsymbol{x}_b$. If the masking distribution during training is uniform [14], and the training loss (Equation (1)) is close to zero, then one can decode in any order using the learned marginals. This gives the uniform unmasking strategy, where given the pre-determined number of decoding steps n, and maximum output sequence length L, a subset of positions $\mathbb{J} \subseteq \mathbb{O}_{\boldsymbol{b}}$ is selected uniformly at random such that $|\mathbb{J}| = \lceil \frac{L}{n} \rceil$, and the tokens are sampled from $p_\theta(v \mid \boldsymbol{x_b}, i)$ for $i \in \mathbb{J}$. However, it has been shown that if the training uses uniform masking, the model cannot be learned well due to the presence of *hard* subproblems [5]. Therefore, in practice using some known order or a heuristic based decoding order works much better than random order decoding [5, 15], which brings us to the sort and select style sampling algorithms.

Algorithm 1 One step of order and select unmasking algorithm

Require: Masked sequence $\boldsymbol{x_b}$, model p_θ, heuristic $\mathcal{O}$, selection algorithm $\mathcal{A}$

Initialize $\boldsymbol{z} \leftarrow \boldsymbol{x_b}$
Forward pass to compute $p_\theta(\cdot \mid \boldsymbol{z})$
Produce heuristic order $\boldsymbol{o} \leftarrow \mathcal{O}(\boldsymbol{z}, p_\theta)$
Choose a subset of masked positions $\mathbb{J} \leftarrow \mathcal{A}(\boldsymbol{x_b}, p_\theta, \boldsymbol{o})$
for each $i \in \mathbb{J}$ **do**
    Sample $\hat{x}_i \sim p_\theta(v \mid \boldsymbol{x_b}, i)$
    Set $z_i \leftarrow \hat{x}_i$
end for
return Updated sequence $\boldsymbol{z}$ (with fewer masked positions)

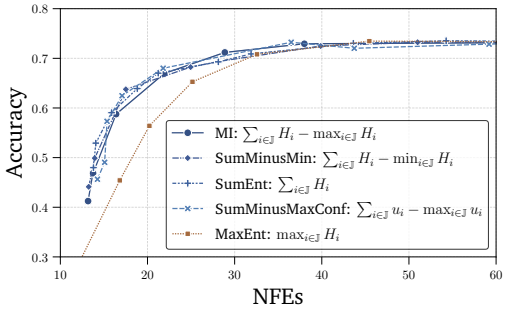

Figure 1: Accuracy vs. Number of function evals (NFEs) for LLaDA-8B-Base on GSM8K. Comparison of different position selection criteria.

Sort and select unmasking. As we increase the number of positions to decode in each step, we naturally incur increasing error by ignoring the dependencies between the selected positions. The total prediction error can be broken down into modeling error and jointness error [3, 4]. Using KL divergence as the divergence measure, the error incurred by ignoring the dependencies between positions while unmasking is simply the mutual information between the joint and margins under the model. The error of selecting a subset $\mathbb{J}$ of masked positions to unmask in one step using the marginals of the model is

$$f_{\text{err}}(\boldsymbol{z}, \mathbb{J}) = \mathop{\mathbb{E}}_{\boldsymbol{v} \sim \mathrm{P}_\theta} \log \frac{\mathrm{P}_\theta(\boldsymbol{v} \mid \boldsymbol{z}, \mathbb{J})}{\prod_{i \in \mathbb{J}} p_\theta(v_i \mid \boldsymbol{z}, i)}$$

where $\mathrm{P}_\theta(\boldsymbol{v} \mid \boldsymbol{z}, \mathbb{J})$ denotes the joint distribution of the selected positions $\mathbb{J}$ given the masked sequence $\boldsymbol{z}$ and $\prod_{i \in \mathbb{J}} p_\theta(v \mid \boldsymbol{z}, i)$ the product of the marginals. Computing an estimate of this error requires multiple forward passes, but as shown in [3], we can get an upper bound on the error in terms of marginal entropies, which can be computed using a single forward pass (see Section A.2):

$$f_{\text{err}}(\boldsymbol{z}, \mathbb{J}) \leq \sum_{i \in \mathbb{J}} H_i - \max_{i \in \mathbb{J}} H_i \tag{2}$$

where, H_i is the marginal entropy of the i-th position, i.e., $-\sum_{v \in \mathbb{V}} p_\theta(v \mid \boldsymbol{z}, i) \log p_\theta(v \mid \boldsymbol{z}, i)$, and the inequality follows from Proposition 1 in Appendix A.2. Using this bound, we can select a subset of masked positions $\mathbb{J}$ in each step [3]:

$$\mathbb{J}^* = \arg\max_{\mathbb{J} \subseteq \mathbb{O}_{\boldsymbol{z}}} |\mathbb{J}|, \quad \text{such that } f_{\text{err}}(\boldsymbol{z}, \mathbb{J}) \leq \epsilon. \tag{3}$$

A tractable greedy embodiment of the above objective is a two step order and select approach: (1) produce a heuristic order of the positions to update, and (2) select a subset of positions to update

by adding one element at a time to the subset according to the provided order. Ordering of the positions implicitly produces an order of the subsets of $\mathbb{O}_{\boldsymbol{z}}$, wherein each subsequent subset has one more position than the previous subset. Ordering the subsets of $\mathbb{O}_{\boldsymbol{z}}$ allows us to avoid trying all possible subsets of $\mathbb{O}_{\boldsymbol{z}}$. Let $\mathcal{O}(\boldsymbol{z}, p_\theta)$ denote the ordering function that returns an ordered list of positions with elements from $\mathbb{O}_{\boldsymbol{z}}$, and $\mathcal{A}(\boldsymbol{z}, p_\theta, \boldsymbol{o})$, the selection criterion that takes in the order and produces the subset of positions to unmask. One step of the decoding using these two functions is shown in Algorithm 1. This approach, which we refer to as "sort and select" style unmasking, can be generalized by replacing $f_{\text{err}}(\boldsymbol{z}, \mathbb{J})$ with other error estimates.

The sort and select is a smooth way to trade off accuracy and throughput. While the error bound Equation (2), which we call MI-bound, is quite natural, one could device other similar position selection criteria. Therefore, before moving further, we perform such a comparison. Specifically, we compare Equation (2) with similar criteria, namely, SumEnt, SumMinusMin, and MaxEnt. We also include SumMinusMaxConf, which in spirit is similar to MI-bound, but instead of marginal entropies, it uses the complement of the confidence, i.e., $1 - p(\hat{x}_i \mid \boldsymbol{z}, i)$. The last criteria in the ablation is MaxEnt, which uses a direct threshold on the marginal entropies, instead of using any accumulation. As seen in Figure 1, which shows the accuracy vs number of function evaluations (NFEs) for LLaDA-8B-Base on GSM8K, the key attribute essential for a good selection criterion is accumulation of uncertainty. The exact form of accumulation does not make a substantial difference. Specifically, MI, SumEnt and SumMinusMin, all perform similarly, whereas direct thresholding using MaxEnt has lower rate of increase in accuracy. We use the MI in the rest of our experiments, but one could potentially use any other criteria that accumulates uncertainty information.

4 Position Contrastive Guidance

We will use a concrete example to motivate the proposed approach of Position Contrastive Guidance. The dark brown bars in the first part of Figure 2 show the probabilities of the top scoring tokens at various masked positions during the first decoding step for LLaDA-8B-Base on an example from GSM8K, i.e., it shows $p_\theta(\hat{x}_i \mid \boldsymbol{z}, i)$, where $\hat{x}_i = \arg\max_{v \in \mathbb{V}} p_\theta(v \mid \boldsymbol{z}, i)$, with the token $\hat{x}_i$ labeled on the x-axis.

We can see that the probability of the space token ␣ is quite high at positions on the right end, almost as high as some of the content tokens at earlier positions. We can be certain that predicting all the consecutive space tokens is erroneous. This problem only gets worse as we go further right. In order to avoid selecting the overconfident positions from the extreme right end, we propose soft left-to-right ordering heuristic (SLR). If $\mathbb{O}_{\boldsymbol{z}}$ denotes the ordered sequence of masked positions, then let k_i be the index of position i in $\mathbb{O}_{\boldsymbol{z}}$. The SLR bias is incorporated into the ordering function $\mathcal{O}$ as:

$$\mathcal{O}(\boldsymbol{z}, p) := \underset{i \in \mathbb{O}_{\boldsymbol{z}}}{\arg\operatorname{sort}}\, \sigma_c(i, \boldsymbol{z}) \ \max_{v}\ p(v \mid \boldsymbol{z}, i),$$

where $\sigma_c(i, \boldsymbol{z}) = \frac{1}{(1+e^{-(c-k_i)/\tau})}$ is the inverted logistic function centered at c that starts at 1 and decays to 0 as i increases. The σ acts like a soft sliding window that biases the ordering to be left-to-right. Note that k_i only increments at masked positions.

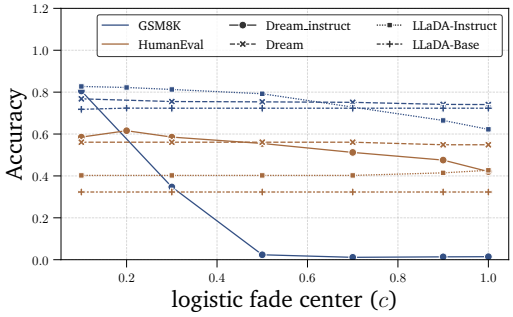

Figure 3: Accuracy vs. SLR bias. Algorithm 1 is executed w and w/o the SLR bias σ (c.f. Equation (4)) in the ordering function $\mathcal{O}$. The SLR bias is varied by changing how far the center of the logistic function is from the leftmost masked position. The x-axis is normalized by the maximum generation length, which varies across the datasets.

Just biasing the decoding order to be left-to-right is not enough. In Figure 2, we can also see that the probability of the space token ␣ at positions 7 or 8 is as high as some of the content tokens at earlier positions like x_4 =6. Since the score of a correct token is close to that of a potentially incorrect token at another position, a sort and select approach would either lead to poor predictions, or would require overly conservative selection threshold. In this specific example, the result of a naive application of the sort and select approach from Algorithm 1 is depicted in the bottom part of

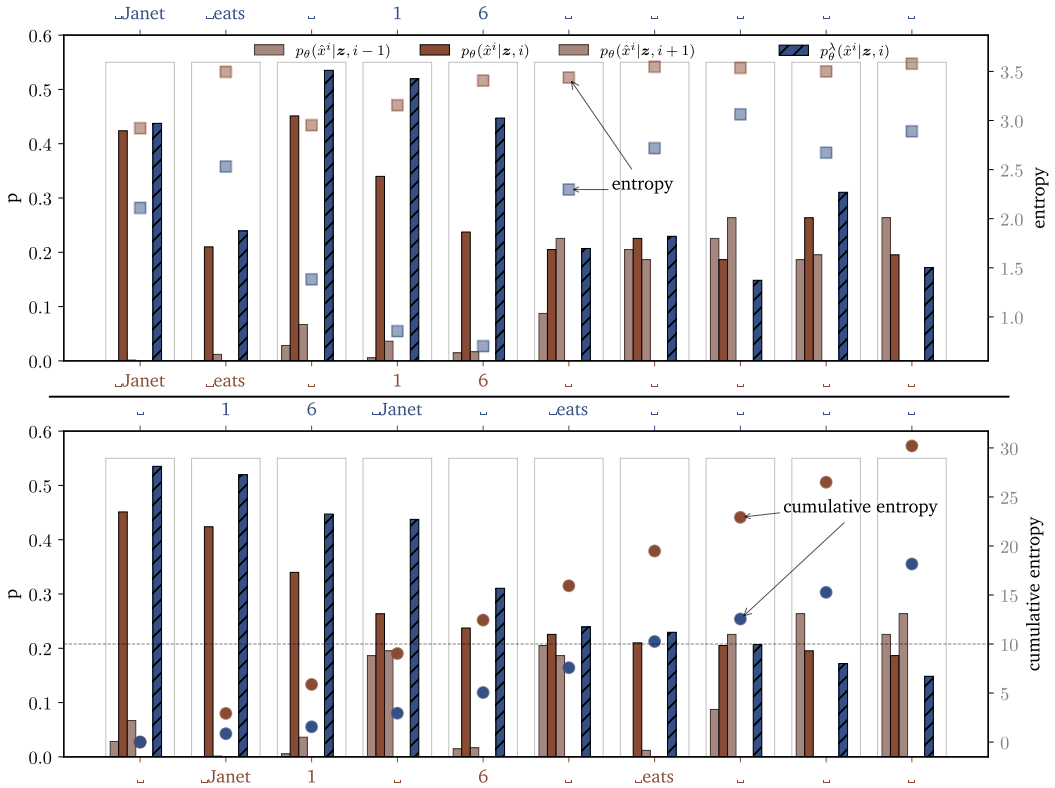

Figure 2: $\boldsymbol{z}$ is the tokenized input sequence "`Q: Janet's ducks lay 16 eggs per day ..... How much in dollars does she make every day at the farmers' market? A:`", followed by all masks. **[Top]** $\hat{x}_i = \arg\max_{v \in \mathbb{V}} p_\theta(v \mid \boldsymbol{z}, i), i \in [\![10]\!]$ is the top scoring token at position i, for the positions that follow the prompt. The **dark brown bars** in the first plot (top) show the probabilities $p_\theta(\hat{x}_i \mid \boldsymbol{z}, i)$, with the token $\hat{x}_i$ labeled on the x-axis. The **light brown bars** show the probabilities $p_\theta(\hat{x}_i \mid \boldsymbol{z}, j)$, for $j \in \{i-1, i+1\}$, i.e., the probability of the same token but at a neighboring position. The **blue bars** show the probabilities of the top token after applying PCG: $p_\theta^{\lambda,w}(\hat{x}_i \mid \boldsymbol{z}, i)$, for $\lambda = 1, w = 1$. The points ■, ■, denote the marginal entropies H_i and $H_i^{\lambda,w}$, respectively. Observe that that the confidence for numerical tokens, like `6`, is lower than some of the ␣ tokens on the right. However, after applying PCG, the confidence for token `6` is higher. **[Bottom]** The plot at the bottom displays the positions after sorting w.r.t the top token probability $p_\theta(\hat{x}_i \mid \boldsymbol{z}, i)$, and $p_\theta^{\lambda,U}(\hat{x}_i \mid \boldsymbol{z}, i)$. The respective tokens are displayed on the x-axis labels below and above the plot, and the points ●, ●, are the cumulative entropies. With a threshold $\epsilon = 10.0$ shown as the horizontal line, the resulting decoding step is `Janet [m] ␣ 1 [m] [m] ␣ [m]...` and `Janet eats ␣ 1 6 ␣ [m]...`, respectively.

Figure 2 wherein we get `Janet [m] ␣ 1 [m] [m] ␣ [m]...` as the output of the decoding step. Now observe the light brown bars in Figure 2. They depict $p_\theta(\hat{x}_i \mid \boldsymbol{z}, i-1)$ and $p_\theta(\hat{x}_i \mid \boldsymbol{z}, i+1)$, i.e., the probability of the top scoring token $\hat{x}_i$ in its neighboring positions. We can see that the model expresses uncertainty by giving similar probabilities to the same token at different positions. For example, the space token $\hat{x}_8 =$ ␣ also has a high probability at $i = 7$ and $i = 9$, but the same is not true for the number token $\hat{x}_4 =$ `6`. Based on the previous observation, we use a guidance signal that utilizes position uncertainty to construct the reference distribution

$$\tilde{p}_\theta^U(v \mid \boldsymbol{z}, i) = \mathop{\mathbb{E}}_{i \sim U(i)} p_\theta(v \mid \boldsymbol{z}, i).$$

We experiment with two different choices of U:

1. **Local average**: $U(i) = \text{Uniform}\{j \in [\![n]\!] \mid |i-j| \leq w, j \notin \mathbb{O}_{\boldsymbol{b}}\}$, i.e., the average of the probabilities of the token v at positions in the window w around i that are masked.

2. **Global average**: $U(i) \propto (1 - \delta_{\mathbb{I}}(i))\, H_i$, where the set $\mathbb{I}$ is the set of positions that were left as masked in the previous decoding step. The idea here is to construct a reference distribution using all the masked positions.

The modified distribution is given as

$$p_\theta^{\lambda,U}(v \mid \boldsymbol{z}, i) \propto \frac{\left(p_\theta(v \mid \boldsymbol{z}, i)\right)^{1+\lambda}}{\left(\tilde{p}_\theta^{U}(v \mid \boldsymbol{z})\right)^{\lambda}}. \tag{4}$$

Overall, PCG can be added to any sort and select unmasking algorithm by adding SLR to $\mathcal{O}$ and using $p_\theta^{\lambda,U}$ instead of p_θ in the selection step $\mathcal{A}$.

5 Experiments

We perform few-shot evaluation on two reasoning datasets, GSM8K [16] and Math500 [17], and a coding dataset, HumanEval [18]. We use the open weights masked diffusion LLMs LLaDA-8B-Base, LLaDA-8B-Instruct [1], Dream-7B, and Dream-7B-Instruct [2] for our experiments. As mentioned above, we use the MI (or the EB) sampler [3] as the base sort and select sampler (Algorithm 1). Note that this base algorithm itself is quite strong and provides a 2-4x speedup over fixed step-sized greedy decoding as noted in Ben-Hamu et al. [3]. Decoding algorithms for autoregressive language models check for certain phrases to appear in order to terminate the generation. For example, ``<|endoftext|>'' or ``\n\nQ:\n'', etc. We implement a similar logic for MDMs, with an additional constraint that all the mask tokens appearing before the early exit phrase are all also filled. To do this we implement a tensorized early exit check that can check all the sequences in a batch against a set of tokenized early exit phrases without converting the token ids into text.

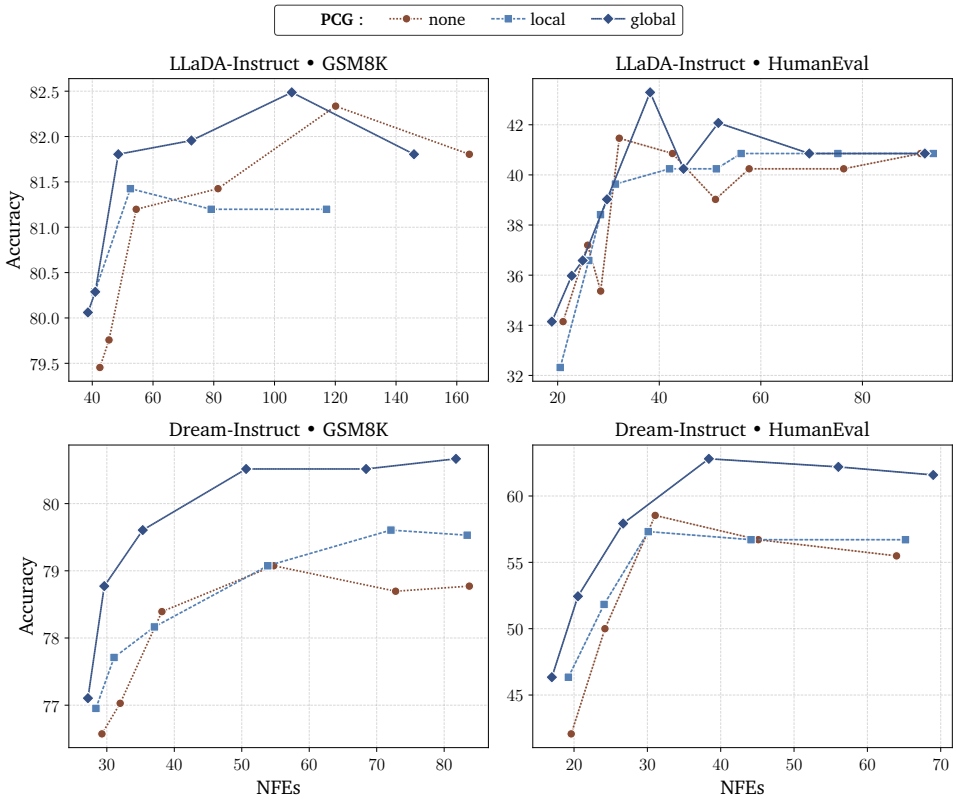

Figure 4: Accuracy vs NFEs for the instruct models as we vary the the selection threshold ϵ. The global PCG generally performs better than the local PCG and no PCG.

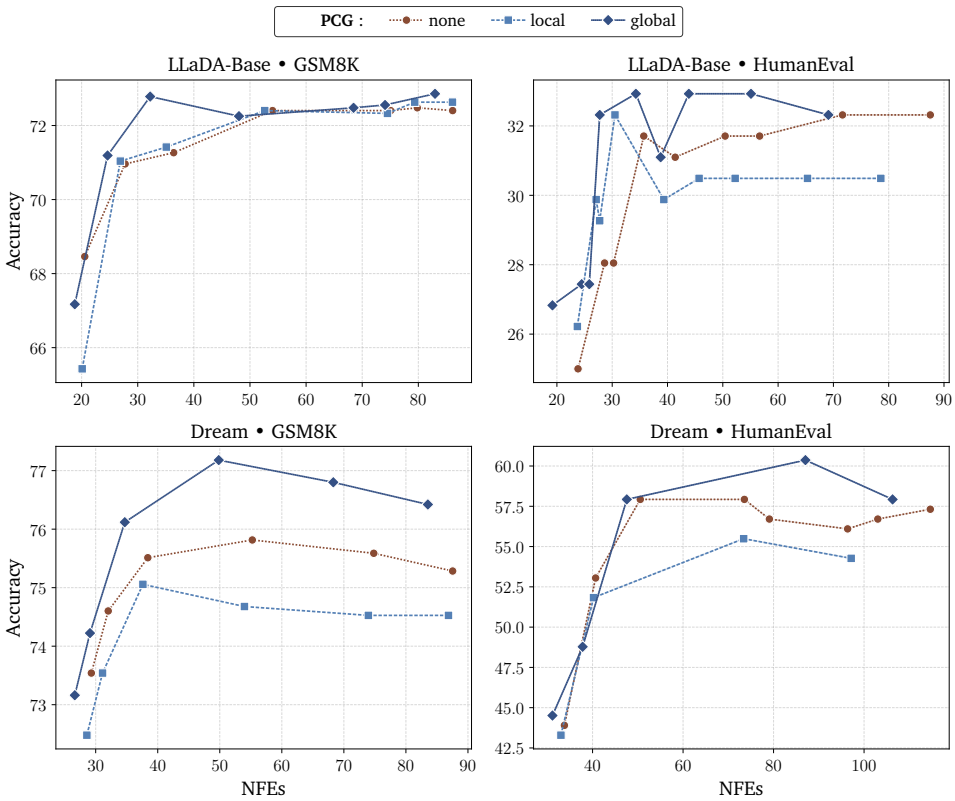

Figure 5: Accuracy vs NFEs for the base model as we vary the the selection threshold ϵ. The global PCG generally performs better than the local PCG and no PCG.

The impact of SLR bias. As showing Figure 3, the SLR bias has a significant impact on the accuracy of both LLaDA-8B-Instruct and Dream-7B-Instruct, but not so much on the base models. Specifically, as the center of the logistic SLR bias is moved to further right, the accuracy for LLaDA-8B-Instruct and Dream-7B-Instruct drop from around 80% on GSM8K to 60% and 5% respectively. While the base models are pre-trained on packed sequences, the instruct models are trained on variable length sequences and are also trained to predict the PAD/EOS tokens. This makes the instruct models overconfident on the positions on the extreme right, which necessitates the use of SLR bias at inference time. Informed by these results, from here onwards we use light SLR bias (equivalent to normalized $c = 0.5$) for the base models, and strong SLR bias ($c = 0.1$) for the instruct models.

Impact of contrastive guidance. In order to get a complete picture of the impact of contrastive guidance on the whole Pareto frontier of accuracy vs speed, we vary the selection threshold ϵ for the MI sampler and ablate the two strategies (global and local). For this set of experiments, we keep the SLR bias on as described above. As seen in Figure 4, the global average generally performs better than the local average reference distribution as well as no guidance. We also observe higher variance in the accuracy on HumanEval due to is small size (164 examples) as compared to GSM8K (1.32k examples). Interestingly, the improvement due to PCG is much larger in the DREAM models, which are trained by adapting an auto-regressive base model, compared LLaDA, which is trained from scratch. Furthermore, as seen in Figure 5, the impact of PCG is smaller on the base models compared to the instruct models in Figure 4. Table 1 reports the final results for the PCG with common hyperparameter values (c and ϵ) for all models and datasets. We find that the PCG consistently improves the accuracy as well as the speed across all the models and datasets, except for the base models on Math500, were we observe a slight degradation in the accuracy.

Table 1: Accuracy and number of forward passes (NFEs). The first row for each dataset reports the results of the MI sort and select (Algorithm 1) with threshold $\epsilon = 0.01$ and SLR bias ($c = 0.1$, 0.5 for instruct and base models, respectively) and the final row reports the results with the addition of classifier-free contrastive guidance using the global average reference distribution. In the round brackets, are the percentage changes in accuracy and NFEs compared to the first row. Entries for which the NFE decreases (resp. increases) are highlighted in green (resp. red), and the same for accuracy, but with the colors inverted.

σ	λ	$\mathcal{A}$	LLaDA-8B-Base		Dream-7B		LLaDA-8B-Instruct		Dream-7B-Instruct	
			Acc(Δ%)	NFE(Δ%)	Acc(Δ%)	NFE(Δ%)	Acc(Δ%)	NFE(Δ%)	Acc(Δ%)	NFE(Δ%)
GSM8K (4-shot, 256 max-length)										
✓		MI	72.4	75	75.6	75	82.3	120	78.7	73
✓	✓	MI	72.5 (↑0)	68 (↓9)	76.8 (↑2)	68 (↓9)	82.5 (↑0)	106 (↓12)	80.5 (↑2)	68 (↓6)
HumanEval (0-shot, 512 max-length)										
✓		MI	32.3	72	56.1	96	40.2	76	55.5	64
✓	✓	MI	32.9 (↑2)	55 (↓23)	60.4 (↑8)	87 (↓10)	40.9 (↑2)	70 (↓9)	62.2 (↑12)	56 (↓12)
MATH500 (4-shot, 512 max-length)										
✓		MI	32.6	153	38.8	141	42.0	264	42.4	311
✓	✓	MI	32.2 (↓1)	136 (↓12)	37.2 (↓4)	129 (↓9)	43.6 (↑4)	229 (↓13)	42.0 (↓1)	269 (↓13)

6 Discussion

We proposed Position Contrastive Guidance (PCG), which uses positional uncertainty to improve the sampling from MDMs. The empirical evaluation suggests that both the components, the soft left-to-right bias and the classifier-free guidance, boost the performance of sort and select style sampling algorithms in terms of accuracy and throughput. Our observations also raise some deeper concerns about the training objectives for MDMs for variable length sequences. Specifically, naive finetuning of MDMs on variable length sequences can lead to problems during inference that necessitate heuristic based fixes like the SLR bias.

Limitations. The empirical evaluation is limited to reasoning and coding tasks. It will be interesting future work to see the impact of PCG open ended generation tasks as well as to explore it use alongside other base sampling algorithms.

References

[1] 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, February 2025. URL http://arxiv.org/abs/2502.09992.

[2] Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7b, 2025. URL https://hkunlp.github.io/blog/2025/dream.

[3] Heli Ben-Hamu, Itai Gat, Daniel Severo, Niklas Nolte, and Brian Karrer. Accelerated sampling from masked diffusion models via entropy bounded unmasking, 2025. URL https://arxiv.org/abs/2505.24857.

[4] 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, 2025. URL https://arxiv.org/abs/2505.22618.

[5] Jaeyeon Kim, Kulin Shah, Vasilis Kontonis, Sham M. Kakade, and Sitan Chen. Train for the Worst, Plan for the Best: Understanding Token Ordering in Masked Diffusions. In *Forty-Second International Conference on Machine Learning*, June 2025. URL https://openreview.net/forum?id=DjJmre5IkP&referrer=%5BReviewers%20Console%5D(%2Fgroup%3Fid%3DICML.cc%2F2025%2FConference%2FReviewers%23assigned-submissions).

[6] Lin Zheng, Jianbo Yuan, Lei Yu, and Lingpeng Kong. A reparameterized discrete diffusion model for text generation, 2024. URL https://arxiv.org/abs/2302.05737.

[7] Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured Denoising Diffusion Models in Discrete State-Spaces. In *Advances in Neural Information Processing Systems*, November 2021. URL https://openreview.net/forum?id=h7-XixPCAL.

[8] Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax Flows and Multinomial Diffusion: Learning Categorical Distributions. In *Advances in Neural Information Processing Systems*, November 2021. URL https://openreview.net/forum?id=6nbpPqUCIi7.

[9] Haoran Sun, Lijun Yu, Bo Dai, Dale Schuurmans, and Hanjun Dai. Score-based Continuous-time Discrete Diffusion Models. In *The Eleventh International Conference on Learning Representations*, September 2022. URL https://openreview.net/forum?id=BYWWwSY2G5s.

[10] Andrew Campbell, Joe Benton, Valentin De Bortoli, Tom Rainforth, George Deligiannidis, and Arnaud Doucet. A Continuous Time Framework for Discrete Denoising Models. In *Advances in Neural Information Processing Systems*, October 2022. URL https://openreview.net/forum?id=DmT862YAieY.

[11] Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 32819–32848. PMLR, July 2024. URL https://proceedings.mlr.press/v235/lou24a.html.

[12] Subham Sekhar Sahoo, Marianne Arriola, Aaron Gokaslan, Edgar Mariano Marroquin, Alexander M. Rush, Yair Schiff, Justin T. Chiu, and Volodymyr Kuleshov. Simple and Effective Masked Diffusion Language Models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, November 2024. URL https://openreview.net/forum?id=L4uaAR4ArM&referrer=%5Bthe%20profile%20of%20Volodymyr%20Kuleshov%5D(%2Fprofile%3Fid%3D~Volodymyr_Kuleshov1).

[13] 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. In *The Thirteenth International Conference on Learning Representations*, October 2024. URL https://openreview.net/forum?id=sMyXP8Tanm.

[14] Kaiwen Zheng, Yongxin Chen, Hanzi Mao, Ming-Yu Liu, Jun Zhu, and Qinsheng Zhang. Masked Diffusion Models are Secretly Time-Agnostic Masked Models and Exploit Inaccurate Categorical Sampling. In *The Thirteenth International Conference on Learning Representations*, October 2024. URL https://openreview.net/forum?id=CTC7CmirNr.

[15] Lin Zheng, Jianbo Yuan, Lei Yu, and Lingpeng Kong. A Reparameterized Discrete Diffusion Model for Text Generation, February 2024. URL http://arxiv.org/abs/2302.05737.

[16] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021.

[17] Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step, 2023. URL https://arxiv.org/abs/2305.20050.

[18] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex

Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

A Appendix

A.1 Summary of Notation

Notation	Description
	General
$\boldsymbol{X}$, $\boldsymbol{x}$	Random variables (capital letters)
$\boldsymbol{x}$, $\boldsymbol{b}$, d	Values of random variables (lowercase)
$\mathbb{X}$, $\mathbb{B}$	Sets (blackboard font)
$\mathbb{B}_n$	Set of all bit vectors of length n, i.e. $\{0,1\}^n$
$[\![n]\!]$	Set of natural numbers $\{1,2,\ldots,n\}$
x_i, X_i	i-th component of $\boldsymbol{x}$ and $\boldsymbol{X}$ respectively
	Specific variables
$\mathbb{V}$	Vocabulary of tokens excluding the mask token
$\mathbb{V}^n$	Set of token sequences of length n, i.e., $\mathbb{V}^n = \mathbb{V} \times \overset{n \text{ times}}{\cdots} \times \mathbb{V}$
v_{mask}	Mask token
$\boldsymbol{b}$	A bit vector of length n, i.e., an element of $\{0,1\}^n$. Used to identify the positions of where the token is unknown (masked). $b_i = 0$ implies i-th token is unknown.
$\bar{\boldsymbol{b}}$	Flipped bit vector of $\boldsymbol{b}$, i.e., $\bar{\boldsymbol{b}} = 1 - \boldsymbol{b}$
$\boldsymbol{x}_b$	**Set** of sequences whose entries match that of $\boldsymbol{x}$ at positions where $\boldsymbol{b}$ is 1. This set can be compactly represented as a sequence in $(\mathbb{V} \cup \{v_{\text{mask}}\})^n$ by replacing the positions where $\boldsymbol{b}$ is 0 with v_{mask}.
$\bar{\mathbb{V}}^n$	$\bar{\mathbb{V}}^n := (\mathbb{V} \cup \{v_{\text{mask}}\})^n$
$\boldsymbol{z}$	Denotes a generic element of $\bar{\mathbb{V}}^n$, without explicitly specifying $\boldsymbol{b}$ and $\boldsymbol{x}$.

Table 2: Summary of notation used throughout the paper.

A.2 Sort and Select Sampling

Proposition 1 justifies the use of sort and select style sampling from MDMs. Specifically, it shows that the mutual information is bounded by sum of entropies at various position minus the maximum entropy.

Proposition 1. *For random variables X, Y, Z with joint distribution $P_{X,Y,Z}$ and marginal distributions P_X, P_Y, P_Z, we have*

$$I(X,Y,Z) \leq H(X) + H(Y) + H(Z) - \max\{H(X), H(Y), H(Z)\}.$$

Proof. The proof which is quite elementary and is similar to [3], is provided here for the sake of completeness.

$$\begin{aligned}
H(X,Y,Z) &= -\sum_{x,y,z} P_{X,Y,Z}(x,y,z) \log P_{X,Y,Z}(x,y,z) \\
&= -\sum_{x,y,z} P_{X,Y,Z}(x,y,z) \log P_{X,Y|Z}(x,y|z) - \sum_{x,y,z} P_{X,Y,Z}(x,y,z) \log P_Z(z) \\
&= H(X,Y|Z) + H(Z) \\
&\geq H(Z) \;\text{ because } H(X,Y|Z) \geq 0
\end{aligned}$$

Same inequality holds for $H(X)$ and $H(Y)$, mutatis mutandis. Therefore, we have $H(X,Y,Z) \geq \max\{H(X), H(Y), H(Z)\}$. Now, by the definition of mutual information, we have

$$\begin{aligned} I(X,Y,Z) &= \sum_{x,y,z} P_{X,Y,Z}(x,y,z) \left(\log P_{X,Y,Z}(x,y,z) - \log P_X(x) - \log P_Y(y) - \log P_Z(z)\right) \\ &= H(X) + H(Y) + H(Z) - H(X,Y,Z) \\ &\leq H(X) + H(Y) + H(Z) - \max\{H(X), H(Y), H(Z)\} \text{ by the inequality above} \end{aligned}$$

□