On-Policy Self-Distillation for Efficient Diffusion Language Models with Early-Stage Calibration

Huaisheng Zhu[†], **MingYu Liu, Junze Liu, Zhen Ge, Tian Wang, Jiri Gesi, Dakuo Wang**
Weiqi Zhang, Houyu Zhang, Yufan Guo, Xian Li, Bing Yin, Sujay Sanghavi

Amazon
hvz5312@psu.edu
{ethenliu,junzeliu,zge,wangtan,jirigesi,dakuow,}
{zhaweiqi,zhanhouy,gyufan,xianlee,alexbyin,sujayrs}@amazon.com

Abstract

Diffusion Large Language Models (DLLMs) have recently achieved strong performance, e.g., masked diffusion models (MDMs) can surpass autoregressive models (ARMs) in various tasks. However, DLLMs often struggle with inaccurate early-stage predictions due to limited context, which hinders both the model's inference efficiency and the output's overall quality. We propose Calibrated On-Policy Self-Distillation (COPSD) for DLLMs, a simple and efficient method to calibrate early token predictions without requiring demonstration data. COPSD distills an unnormalized target distribution derived from later decoding steps into the original model, enabling more accurate early predictions during inference. Experiments on math, planning, and RLHF tasks show that COPSD improves both effectiveness and efficiency, and further enhances performance when combined with supervised fine-tuning.

1 Introduction

Recently, Diffusion Large Language Models (DLLMs) have made remarkable progress in various tasks (Sohl-Dickstein et al., 2015; Austin et al., 2021; Campbell et al., 2022; Lou et al., 2023; Meng et al., 2022). By optimizing the evidence lower bound or its simplified variants, masked diffusion models (MDMs) have achieved performance comparable to, and in some cases surpassing, that of autoregressive models (Sahoo et al., 2024; Shi et al., 2024; Nie et al., 2025). Moreover, recent studies have investigated the scaling properties of MDMs, demonstrating that they can achieve competitive performance with state-of-the-art autoregressive models of similar size (e.g., LLaMA 2 (Touvron et al., 2023) and LLaMA 3 (Dubey et al., 2024)) across a range of downstream tasks (Nie et al., 2025; Gong et al., 2024; Nie et al., 2024; Gong et al., 2025; Sahoo et al., 2025; Zhu et al., 2025b).

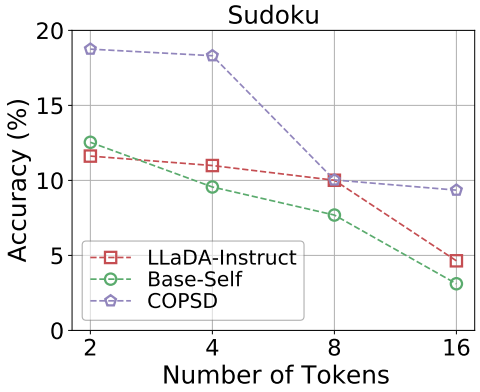

Figure 1: Performance comparison of different baselines using the same base model (LLaDA-Instruct) based on different number of tokens generated simultaneously.

Despite these advancements, most existing work on DLLMs has focused on unsupervised pretraining, supervised fine-tuning, or reinforcement learning-based fine-tuning (Zhu et al., 2025a; Zhao et al., 2025; You et al., 2025). However, DLLMs often suffer from inaccurate token predictions during the early stages of generation, when nearly all tokens are masked and contextual information is limited. In contrast, later stages benefit from a larger number of unmasked tokens, enabling more accurate predictions through richer context. These early-stage inaccuracies not only hurt output quality but also reduce model efficiency, especially since DLLMs typically generate multiple tokens in parallel and rely heavily on accurate contextual cues. As shown in Figure 1, DLLM-based models can accurately predict tokens and achieve correct results when generating two tokens simultaneously, as indicated by the red line. However, generating more tokens at once leads to a noticeable drop in performance, which limits the efficiency of DLLMs. This is likely because the models tend to produce inaccurate predictions in the early stages due to limited context. These early errors can then propagate and adversely impact subsequent predic-

[†]Work done as an intern at Amazon

Findings of the Association for Computational Linguistics: ACL 2026, pages 26954–26965

tions that rely on the earlier outputs.

To address these challenges, this paper proposes a On-Policy Self-Distillation approach for DLLMs that calibrates the prediction distribution of early-stage tokens, called **Calibrated On-Policy Self-Distillation (COPSD)**. Specifically, we introduce an unnormalized target distribution that leverages information from later decoding steps to refine early predictions. However, since such future context is unavailable during inference, we propose to on-policy distill this calibrated distribution into the original DLLM. This encourages the model to develop an inherent ability to produce more accurate early-stage predictions, even without access to future tokens at test time. To stabilize training, we use a group relative normalization method, which helps mitigate the challenges due to the limited number of samples that can be drawn during training.

Our contributions. We propose Calibrated On-Policy Self-Distillation (COPSD), an efficient algorithm that requires no demonstration data and calibrates the prediction distribution at the early stages of generation. By improving early-stage predictions, COPSD enhances both the efficiency and effectiveness of current DLLMs. To the best of our knowledge, this is the first On-Policy Self-Distillation method specifically designed to address early-stage calibration in DLLMs, offering a practical recipe for improving their overall performance.

Experiments. We conduct experiments with COPSD on tasks involving math, planning, and RLHF, and observe consistent improvements over naive self-distillation baselines. Notably, COPSD significantly boosts efficiency on simpler tasks such as Sudoku, enabling the model to generate more tokens in parallel while maintaining performance comparable to baseline models that generate fewer tokens at a time. Furthermore, when combined with supervised fine-tuning, our self-distillation approach further enhances performance, demonstrating the effectiveness and versatility of COPSD.

2 Related Works

Diffusion Language Models. The development of DLLMs is inspired by recent advances in discrete diffusion models, which introduced novel forward and reverse transition mechanisms and enabled a variety of model variants (Sohl-Dickstein et al., 2015; Austin et al., 2021; Campbell et al., 2022; Lou et al., 2023; Meng et al., 2022). Empirical results show that masked diffusion models (MDMs) can achieve perplexity on par with autoregressive models (ARMs) (Sahoo et al., 2024; Shi et al., 2024; Nie et al., 2025; Ou et al., 2024). To improve training efficiency, several works have proposed simplified training objectives for masked diffusion processes with theoretical justifications. In addition, recent research has examined the scaling behavior of MDMs, including both training from scratch and adaptation from pre-trained ARMs (Nie et al., 2025; Gong et al., 2024; Nie et al., 2024).

On-Policy Distillation. Our work connects to On-Policy Distillation methods, which train a student model directly on trajectories sampled from its own policy, while a teacher model provides per-token guidance via KL-based regularization or related objectives (Agarwal et al., 2024; Xu et al., 2024; Gu et al., 2023; Xiao et al., 2026). This approach mitigates distribution shift by optimizing over the student's visitation distribution, but it typically relies on a separate, often larger, teacher model. To eliminate reliance on large teacher models, on-policy self-distillation has been proposed, enabling a model to improve its performance by leveraging its own inherent capabilities (Zhao et al., 2026; Hübotter et al., 2026; Kim et al., 2026).

Training of Diffusion Language Models. Beyond unsupervised pre-training on large-scale datasets, some studies have explored supervised fine-tuning and reinforcement learning-based methods for DLLMs (Zhu et al., 2025a; Zhao et al., 2025; You et al., 2025; Prabhudesai et al., 2025). However, these approaches largely follow training pipelines originally designed for autoregressive models, overlooking the unique inference dynamics of diffusion models—particularly the multi-step unmasking process that generates several tokens in parallel to improve inference speed (Nie et al., 2025). This oversight is critical, as DLLMs often struggle with inaccurate early-stage predictions, which significantly limit their efficiency in parallel generation and degrade overall generation quality. To address these challenges, we propose Calibrated On-Policy Self-Distillation (COPSD), a self-distillation method designed to calibrate the token distribution at early stages of generation. To the best of our knowledge, this is the first work to systematically study the limitations of early-stage token predictions and introduce a On-Policy Self-Distillation framework tailored for Efficient DLLMs.

3 Preliminaries

Let the text sequence $\mathbf{x} = [x_1, x_2, \ldots]$ represent the input prompt, and $\mathbf{y} = [y_1, y_2, \ldots]$ represent the generated response. We denote the DLLM by $p_\theta(\mathbf{y} \mid \mathbf{x})$ for simplicity, where θ are the model parameters and the probability defines the probability of generating response $\mathbf{y}$ conditioned on the $\mathbf{x}$. In this paper, we focus on On-Policy Self-Distillation settings, where we denote the set of prompts in a dataset $\mathcal{D}_p$ as $\{\mathbf{x}_0, \ldots, \mathbf{x}_N\}$, with N representing the total number of samples in the dataset.

Diffusion Language Models. Specifically, given a context $\mathbf{x}$, a DLLM begins generation from an initial response $\mathbf{y}_T$, where all tokens are masked (denoted as M), i.e., $y_T^i = \mathrm{M}$ for all positions i. The model then iteratively unmasks tokens in a reverse diffusion process. At each timestep t, it predicts the distribution over previous states as $p_\theta(\mathbf{y}_{t-1} \mid \mathbf{x}, \mathbf{y}_t)$. During inference, the model selects a subset of n tokens to update, changing their states from masked to unmasked based on the predicted distribution. Mathematically, the update at step $\mathbf{y}_t$ can be expressed as $\mathbf{y}_{t-1}^{\mathcal{I}} \sim p_\theta\left(\mathbf{y}_{t-1}^{\mathcal{I}} \mid \mathbf{x}, \mathbf{y}_t\right)$, where $\mathcal{I}$ is the set of indices of masked tokens in $\mathbf{y}_t$ and the size of $\mathcal{I}$ is n. Then, DLLMs update tokens at those positions, while the rest remain unchanged:

$$\mathbf{y}_{t-1}^i = \begin{cases} \mathbf{y}_{t-1}^i & \text{if } i \in \mathcal{I} \\ \mathbf{y}_t^i & \text{otherwise} \end{cases}. \tag{1}$$

Supervised Fine-tuning. In this paper, we focus on the most widely adopted class of diffusion language models—masked diffusion language models (MDMs)—for supervised fine-tuning. Specifically, fine-tuning a DLLM involves a forward noising process that progressively corrupts an input token sequence $\mathbf{y}_0$ by the mask token. Specifically, at timestep t, the sequence $\mathbf{y}_t$ is partially masked, where for each token the probability of masking is t/T. The process is indexed by time $t \in [0, T]$. Given a partially masked sequence, we follow prior work that adopts simplified objectives derived from the original negative evidence lower bound (NELBO) for training MDMs. Therefore, the supervised fine-tuning objective on each sample can be written as follows (You et al., 2025):

$$\mathcal{L}_{\mathbf{y}_t} = \sum_{j=1}^{|\mathbf{y}_t|} \mathbb{1}\left[\mathbf{y}_t^j = \mathrm{M}\right] \log p_\theta(\mathbf{y}_0^j \mid \mathbf{y}_t, \mathbf{x}), \tag{2}$$

where $|\mathbf{y}_t|$ is the sequence length of $\mathbf{y}_t$ and $\mathbf{y}_0$ is the ground truth response of the prompt $\mathbf{x}$. The key difference between masked DLLMs and BERT (Devlin et al., 2019) lies in their masking and decoding strategies. BERT employs a fixed masking ratio and performs single-step infilling, while masked DLLMs use time-dependent masking ratios and a multi-step decoding process that starts from fully masked inputs and progressively denoises them. This iterative refinement enables masked DLLMs to function as true generative models. The final training objective over the entire dataset with prompt and response pairs can be formulated as:

$$\min_\theta -\mathbb{E}_{t\sim\mathcal{U}\{0,T\},\mathbf{x},\mathbf{y}\sim p_{\text{data}},\mathbf{y}_t\sim q_{t|0}} \mathcal{L}_{\mathbf{y}_t}, \tag{3}$$

where the prompt $\mathbf{x}$ and its corresponding response $\mathbf{y}$ are sampled from the data distribution p_{data} and a timestep t is uniformly sampled from $\{0, 1, \ldots, T\}$, denoted as $\mathcal{U}\{0, T\}$. The forward process $q_{t|0}$ denotes the corruption mechanism that randomly masks tokens in $\mathbf{y}_0$, where each token is independently masked with probability t/T.

Naive Self-Distillation. In this paper, we focus on the Self-Distillation of Diffusion Large Language Models (DLLMs). As our baseline, we adopt a straightforward Self-Distillation approach: we iteratively sample response $\hat{\mathbf{y}}$ for each prompt $\mathbf{x}$ from the DLLM, denoted as $p_\theta(\hat{\mathbf{y}} \mid \mathbf{x})$, and then fine-tune the model by maximizing the likelihood of these sampled responses. Specifically, we optimize the following loss objective for each prompt $\mathbf{x}$ and its corresponding generated response $\hat{\mathbf{y}}$:

$$\mathcal{L}_b = -\sum_{j=1}^{|\hat{\mathbf{y}}_t|} \mathbb{1}\left[\hat{\mathbf{y}}_t^j = \mathrm{M}\right] \log p_\theta(\hat{\mathbf{y}}_0^j \mid \hat{\mathbf{y}}_t, \mathbf{x}). \tag{4}$$

We put details of this baseline into Algorithm 1.

4 Method

In this section, we mainly introduce Calibrated On-Policy Self-Distillation (COPSD), a framework for DLLMs that encourages the model to calibrate its early-stage token distribution. By improving the accuracy of token predictions in the initial decoding steps, COPSD enables DLLMs to generate more reliable outputs earlier in the process. This not only enhances overall prediction quality but also improves efficiency by supporting parallel generation of more tokens during early stages. Specifically, we propose a calibrated distribution that leverages future state information in DLLMs to improve early-stage predictions. To incorporate this information, we introduce a self-distillation framework based on

Algorithm 1 Naive Self-Distillation

Require: Dataset $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\} \in \mathcal{D}_p$ where $\mathbf{x}^{(1)}$ is a prompt, a pretrained diffusion language model $\mathbf{p}_0(\mathbf{y} \mid \mathbf{x})$

1: **for** each training iteration **do**
2: For each prompt $\mathbf{x}$ from $\mathcal{D}$ and then sample $\mathbf{y}$ from $p_0(\mathbf{y} \mid \mathbf{x})$
3: Sample $t \sim \mathcal{U}\{0, T\}$
4: For response $\mathbf{y}$, Mask tokens in $\mathbf{y}$ with probability t/T to obtain $\mathbf{y}_t$
5: Update the parameter θ with $\mathcal{L}_{\text{b}}$
6: **end for**

KL divergence that enables the model to learn from the calibrated distribution in an implicit manner. To ensure training stability, we further apply a group normalization technique for our model training.

4.1 Calibrated Distribution

DLLMs are originally designed to predict tokens in parallel and in arbitrary order, offering the potential for faster inference. However, in practice, they still require a large number of sampling steps to generate high-quality outputs. A key factor contributing to this inefficiency is the inaccuracy of early-stage token predictions. Specifically, DLLMs iteratively unmask tokens until a fully unmasked sequence is produced, meaning that later-stage predictions heavily depend on the quality of earlier ones. A central intuition from prior work is that tokens unmasked at early generation steps tend to have suboptimal probability distributions due to insufficient contextual information. Remasking these early tokens in later steps can substantially improve performance (Nie et al., 2025; Wang et al., 2025), in contrast to tokens unmasked at later steps, which benefit from richer context and thus yield more reliable predictions. Motivated by this intuition, we propose an unnormalized distribution that leverages later steps to calibrate the earlier token prediction distribution $p_\theta(\mathbf{y}^i_{t-1} \mid \mathbf{y}_t, \mathbf{x})$ Specifically, we utilize the output $p_\theta(\mathbf{y}^i_{t-1} \mid \mathbf{y}^i_{t-s}, \mathbf{x})$ which provides more contextual information at a later step $t-s$, where $t-s<t$, and yields more accurate predictions, denoted as $\tilde{p}_\theta$ for simplicity. The calibrated distribution $p(\mathbf{y}^i_{t-1} \mid \mathbf{y}_t, \mathbf{y}_{t-s}, \mathbf{x})$ for the token at the position i is defined as the following Equation:

$$\begin{aligned} &p(\mathbf{y}^i_{t-1} \mid \mathbf{y}_t, \mathbf{y}_{t-s}, \mathbf{x}) \\ =&p_\theta(\mathbf{y}^i_{t-1} \mid \mathbf{y}_t, \mathbf{x}) \cdot \frac{e^{b^i(\mathbf{y}_t, \mathbf{y}_{t-s})}}{\mathbf{Z}}, \end{aligned} \tag{5}$$

where the calibration term is $b^i(\mathbf{y}_t, \mathbf{y}_{t-s}) = -\|\tilde{p}_{\theta'}(\mathbf{y}^i_{t-1} \mid \mathbf{y}_t, \mathbf{x}) - p_{\theta'}(\mathbf{y}^i_{t-1} \mid \mathbf{y}_{t-s}, \mathbf{x})\|_2^2$ measures the distance between two token distributions at the position i: one at timestep t, where fewer tokens are unmasked and contextual information is limited, and the other at timestep $t-s$, where more tokens are unmasked, providing richer context and enabling more accurate token predictions. We use $p_\theta(\mathbf{y}^i_{t-1} \mid \mathbf{y}_t, \mathbf{x})$ to represent the probability and $\mathbf{Z} = \sum_{\mathbf{y}^i_{t-1} \in \mathcal{V}} p_\theta(\mathbf{y}_{t-1} \mid \mathbf{y}_t, \mathbf{x}) e^{b(\mathbf{y}_t, \mathbf{y}_{t-s})}$. We have following properties for this distribution: when earlier time steps show greater inconsistencies, the probability values are reduced. Conversely, if the discrepancies are smaller in earlier steps, the probability values are increased. This approach is founded on the principle that predictions made during later time steps are inherently more accurate due to their access to more contextual tokens (unmasked tokens). As more contextual tokens become available, the model can make more informed and reliable predictions compared to earlier steps where limited context is available. By leveraging this calibrated distribution, we can rectify inaccurate tokens from previous steps using information gained in later steps. This approach enables us to unmask more tokens at earlier stages while maintaining accuracy, as the previous steps' tokens can be corrected through the calibrated distribution. Consequently, this calibrated distribution can potentially reduce the total number of sampling steps and enhance the overall performance of DLLMs. By improving the accuracy of early-stage token predictions, it enables the model to safely generate more tokens in parallel, offering a clear advantage over uncalibrated approaches.

4.2 Calibrated On-Policy Self-Distillation

In the previous section, we introduced a calibrated distribution that can improve both the performance and efficiency of DLLMs. However, applying this distribution directly during inference is challenging, as it relies on future sequence information that is not accessible at test time. To address this, we propose a self-distillation approach that distills the calibrated distribution into the original DLLM using reverse KL divergence, enabling the model to approximate the benefits of future context without explicitly accessing it for the timestep t:

$$\mathbb{D}_{\text{KL}}[p_\theta(\mathbf{y}_{t-1} \mid \mathbf{y}_t, \mathbf{x}) \| p(\mathbf{y}_{t-1} \mid \mathbf{y}_t, \mathbf{y}_{t-s}, \mathbf{x})] \tag{6}$$

However, directly optimizing the reverse KL divergence is either intractable or unstable during train-

ing, as it requires sampling from the model itself. To address this issue, we propose distilling knowledge from earlier optimization steps, where the model parameters are denoted as θ'. During training, we employ a weighted sampling algorithm that leverages the predictions of the model at these previous checkpoints. Specifically, for each timestep t and token position i, we perform weighted sampling based on the outputs of the model with parameters θ'. In doing so, we replace θ with θ' to provide a more stable target distribution, thereby guiding the optimization of the reverse KL divergence in Equation (6) more effectively given $\mathbf{y}_{t-s}$:

$$\mathcal{L}_i^t = \mathbb{E}_{\mathbf{y}_t \sim p_\theta}(\log p_\theta^i(\mathbf{y}_t) - \log p^i(\mathbf{y}_t)) = \mathbb{E}_{\mathbf{y}_t \sim p_{\theta'}} \frac{p_\theta^i(\mathbf{y}_t)}{p_{\theta'}^i(\mathbf{y}_t)} (\log \frac{p_\theta^i(\mathbf{y}_t)}{p_{\theta'}^i(\mathbf{y}_t)} + b^i(\mathbf{y}_t, \mathbf{y}_{t-s})), \quad (7)$$

where $p^i(\mathbf{y}_t)) = p(\mathbf{y}_{t-1} \mid \mathbf{y}_t, \mathbf{y}_{t-s}, \mathbf{x})$, $p_\theta^i(\mathbf{y}_t) = p_\theta(\mathbf{y}_{t-1}^i \mid \mathbf{y}_t, \mathbf{x})$, $p_{\theta'}^i(\mathbf{y}_t) = p_{\theta'}(\mathbf{y}_{t-1}^i \mid \mathbf{y}_t, \mathbf{x})$ and $\tilde{p}_{\theta'}^i(\mathbf{y}_{t-s}) = \tilde{p}_{\theta'}(\mathbf{y}_{t-1}^i \mid \mathbf{y}_{t-s}, \mathbf{x})$ for simplicity. Accurately estimating the $\log \mathbf{Z}$ term requires a large number of samples, which is impractical in real implementations. For unnormalized models such as those used in Noise Contrastive Estimation, a common solution is to treat as a learnable parameter and optimize it jointly with the model (Gutmann and Hyvärinen, 2010). However, this introduces additional computational overhead and effectively requires training an extra parameter or sub-model. To avoid this issue, the referenced work argues that $\mathbf{Z}$ can be safely omitted in practice (Song and Kingma, 2021). Therefore, we adopt this simplification throughout our method.

4.3 Practical Implementation

Although sampling from previous timesteps helps stabilize training, optimizing the loss in Equation (7) still presents several additional challenges. To train each timestep t in Equation (7), we typically perform a reverse diffusion step by iteratively unmasking tokens at that specific timestep. However, this procedure becomes computationally inefficient when sampling multiple timesteps t for the same prompt during training. To address this, we propose an optimized training strategy that combines the reverse step with the forward diffusion process, leveraging pre-sampled data from the original diffusion language model. This integration results in the following training objective:

$$\min_\theta \mathbb{E}_{\substack{\mathbf{y} \sim p_0(\cdot|\mathbf{x}),\ t \sim \mathcal{U}\{0,T\} \\ \mathbf{y}_{t-s} \sim p_{t-s}(\cdot|\mathbf{y})}} \mathcal{L}_i^t, \quad (8)$$

where $p_{t-s}(\cdot \mid \mathbf{y})$ represents the forward diffusion process in DLLMs, where each token in the sequence $(t-s)/T$ is independently masked with probability. From our empirical studies, we find that fixing the timestep t accelerates the training of this objective. Therefore, in our experiments, we keep t fixed during training. Additionally, since $\mathcal{L}_i^t$ depends on $\mathbf{y}_t$using the forward diffusion process. This leads to the following formulation for $\mathcal{L}_i^t$:

$$\mathbb{E}_{\mathbf{y}_t \sim p_t} \frac{p_\theta^i(\mathbf{y}_t)}{p_{\theta'}^i(\mathbf{y}_t)} (\log \frac{p_\theta^i(\mathbf{y}_t)}{p_{\theta'}^i(\mathbf{y}_t)} + b^i(\mathbf{y}_t, \mathbf{y}_{t-s})), \quad (9)$$

where $\mathbf{y}_t$ is also sampled from the forward distribution DLLMs, $p_t(\mathbf{y} \mid \mathbf{x})$.

4.4 Group Relative Normalization

However, training the objective in Equation (9) stably requires sampling a large number of candidates from $p_{\theta'}^i$, which is impractical in real-world implementations. In practice, computational constraints limit the number of samples that can be generated per prompt during training. As a result, optimizing the objective guided by the calibration term can introduce high variance, and we empirically observe that this often leads to training instability or model collapse. To resolve this problem, inspired by techniques in reinforcement learning—such as Group Relative Policy Optimization (GRPO)—we apply a normalization strategy to the calibration term to ensure stable and efficient training. Specifically, for each prompt $\mathbf{x}$, we first sample a set of responses $\{\mathbf{y}^1, \ldots, \mathbf{y}^K\}$from the DLLM using parameters from a previous optimization step, denoted as θ'. We refer to this set as $\mathcal{Y}$. To simplify the notation in the following part, we denote $\|\tilde{p}_{\theta'}(\mathbf{y}_{t-1}^{k,i} \mid \mathbf{y}_{t-s}^k, \mathbf{x}) - p_\theta(\mathbf{y}_{t-1}^{k,i} \mid \mathbf{y}_t^k, \mathbf{x})\|_2^2$ as $b^i(\mathbf{y}_t^k, \mathbf{y}_{t-s}^k)$ for i-th token of k-th response for the prompt $\mathbf{x}$. Also, we define the set of response and its corresponding value for the calibration term at the i-th token is $\mathbf{B}^i = [b^i(\mathbf{y}_t^1, \mathbf{y}_{t-s}^1), \ldots, b^i(\mathbf{y}_t^K, \mathbf{y}_{t-s}^K)]$. To reduce the variance introduced by the calibration term, we apply group normalization following the GRPO framework on the i-th token. The normalized calibration term is defined as:

$$\mathbf{A}^{k,i} = \frac{b^i(\mathbf{y}_t^k, \mathbf{y}_{t-s}^k) - \mathrm{mean}(\mathbf{B}^i)}{\mathrm{std}(\mathbf{B}^i)}, \quad (10)$$

where mean and std are the mean value and standard deviation of the calibration term within groups. We replace the original calibration term at token position i in Equation (7) with a normalized version,

Algorithm 2 Calibrated On-Policy Self-Distillation

Require: Dataset $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\} \in \mathcal{D}_p$ where $\mathbf{x}^{(1)}$ is a prompt, a pretrained diffusion language model $p_0(\mathbf{y} \mid \mathbf{x})$ and fixed t and s

1: **for** each training iteration **do**
2: For each prompt $\mathbf{x}$ from $\mathcal{D}_p$ and then sample $\mathbf{y}$ from $p_0(\mathbf{y} \mid \mathbf{x})$
3: For response $\mathbf{y}$, Mask tokens in $\mathbf{y}$ with probability t/T to obtain $\mathbf{y}_t$,
4: For each response $\mathbf{y}$, Mask tokens in $\mathbf{y}$ with probability $(t-s)/T$ to obtain $\mathbf{y}_{t+s}$
5: Update the parameter θ with $\mathcal{L}_{\text{COPSD}}$
6: **end for**

resulting in the following loss formulation:

$$\tilde{\mathcal{L}}_i^t = \mathbb{E}_{\mathbf{y}_t \sim p_t} \frac{p_\theta^i(\mathbf{y}_t)}{p_{\theta'}^i(\mathbf{y}_t)} (\log \frac{p_\theta^i(\mathbf{y}_t)}{p_{\theta'}^i(\mathbf{y}_t)} + \mathbf{A}^{k,i})). \quad (11)$$

Similar to GRPO, this normalization helps stabilize the training process, as also supported by our empirical findings. Based on this, we propose to optimize the following objective for COPSD:

$$\mathcal{L}_{\text{COPSD}} = \mathbb{E}_{\substack{\mathbf{y} \sim p_0(\cdot|\mathbf{x}),\ t \sim \mathcal{U}\{0,T\} \\ \mathbf{y}_{t-s} \sim p_{t-s}(\cdot|\mathbf{y})}} \tilde{\mathcal{L}}_i^t. \quad (12)$$

We put the details of our algorithm in Algorithm 2.

5 Experiment

In this section, we present the main experiment results with ablation studies and hyperparameter analysis, highlighting the performance of COPSD on human alignment and math or planning tasks.

5.1 Experimental Setup

Datasets. We evaluate our methods on widely used datasets for Math, Planning and human alignment tasks. Specifically, for human alignment tasks, we use the Reddit TL;DR summarization dataset (Völske et al., 2017) and the Anthropic-HH dataset (Bai et al., 2022). For math and planning tasks, we use the MATH (Lightman et al., 2023) and GSM8K (Cobbe et al., 2021) datasets. Additionally, we include 4×4 Sudoku puzzles, which require constraint satisfaction and systematic reasoning to correctly fill the grid with numbers. We use the accuracy of final answer for these tasks' evaluation. Due to the low quality of data from MATH and GSM8k, it's hard to further improve the performance of LLaDA models which may lead to overfitting issues. Therefore, for the SFT experiments in this paper, we use ground-truth data from the S1k dataset (Muennighoff et al., 2025), which provides high-quality supervision. Details of the datasets are provided in Appendix A.1.

Models. For all tasks and all experiments, we conduct experiments using LLaDA-8B-Instruct (Nie et al., 2025) as our base models.

Baselines. We compare our model with base DLLM and self-distillation baseline methods on all tasks. Additionally, for reasoning tasks, we compare COPSD with models fine-tuned through SFT as well as with larger-scale LLMs.

Evaluation and Implementation Details. We evaluate performance on reasoning tasks by the accuracy of final answer for each math problems. For human alignment tasks, we compare the response between the trained models and base models with human evaluation. Specifically, we compare the proportion of responses that outperform those of the base model, referred to as the win rate. The remaining data is used for reinforcement learning. To evaluate performance, we use Claude 3.7 to compare responses from baseline or trained models against those generated by the SFT model, with win rate serving as the primary evaluation metric.

5.2 Comparison on Math and Planning tasks

Table 1 compares the performance of COPSD with other base models and self-distillation baselines, evaluated on the Sudoku, MATH and GSM8k datasets. We conduct experiments using LLaDA-Instruction as backbone models. We firstly observe that COPSD consistently outperforms the base model with different number of sampling tokens for each step. For example, COPSD achieves a 61.4% improvement in relative performance on the Sudoku task when predicting 2 tokens simultaneously, and it shows a similar level of improvement when predicting 4 tokens at once. This is because the models can predict tokens more correctly at the early stage of the sampling step for diffusion models. More interestingly, we observe that our method, COPSD, achieves comparable performance when predicting 16 tokens simultaneously to that of the original base models predicting just 2 tokens, which significantly improves prediction efficiency.This further validates the effectiveness of our method, which aligns the distribution of early-step token predictions based on limited context with that of later-step predictions that leverage richer contextual information. By aligning these distributions, our approach enables more parallel token predictions, improving DLLM efficiency.

Table 1: **Model performance on Mathematics and Planning Benchmarks: Green values** indicate best performance . The results demonstrate that **COPSD** consistently outperforms other models, applying consistently improves the starting checkpoint, and alone shows better performance than SFT.

	Sudoku				MATH500				GSM8k			
Model / # Tokens	**2**	**4**	**8**	**16**	**2**	**4**	**8**	**16**	**2**	**4**	**8**	**16**
LLaDA-8B-Instruct	11.62	10.99	10.01	4.64	32.40	28.00	**18.20**	12.00	78.92	69.98	35.10	14.10
Base-Self	12.54	9.56	7.69	3.11	28.60	23.00	17.20	**13.80**	77.89	67.23	36.48	10.01
COPSD	**18.75**	**18.31**	**10.03**	**9.35**	**34.60**	**29.20**	18.00	12.20	**80.74**	**73.24**	**40.66**	**15.77**
LLaDA-8B-Instruct + SFT on S1k	14.11	13.87	3.81	1.57	33.4	24.60	10.80	5.00	79.45	66.49	17.51	1.82
+ Base-Self	14.56	11.05	2.01	0.58	29.40	24.40	13.00	6.20	78.54	68.64	21.61	3.71
+ COPSD	**19.48**	**19.32**	**14.75**	**4.25**	**35.00**	**26.60**	**16.80**	**6.60**	**80.74**	**70.03**	**28.20**	**4.70**

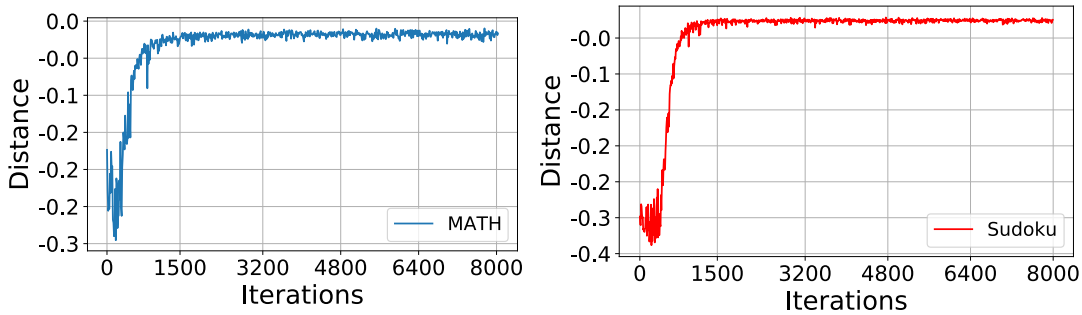

Figure 2: Training Dynamics of Token Distribution Distance for MATH and Sudoku, where the y axis uses $b^i(\mathbf{y}_t, \mathbf{y}_{t-s})$ in Equation (5) to measure the distance gap between early and later tokens' distributions.

Moreover, to further evaluate the effectiveness of our method, we incorporate standard fine-tuning techniques such as Supervised Fine-Tuning (SFT) using ground-truth data. We first observe that training on high-quality reasoning data improves performance on math and planning datasets when predicting two tokens simultaneously across all benchmarks. However, when predicting more tokens at once, the performance tends to decline compared to the original base models. Notably, when combined with SFT, we find that our method consistently enhances performance across various numbers of simultaneously predicted tokens. This verifies the potential of COPSD combined with SFT.

Finally, we present the training dynamics of COPSD with respect to the distance $b^i(\mathbf{y}_t, \mathbf{y}_{t-s})$ used in Equation (5), which is shown in Figure 2. These dynamics demonstrate that our model consistently reduces the distance between early and later timesteps, highlighting both the effectiveness and the characteristic training behavior of our method.

5.3 Comparison on Human Alignment tasks

Table 2 presents a comparison of win rates between our base models and various baselines on human alignment and summarization tasks. Specifically, we use the base model configured to generate two tokens at a time as the reference. We then evaluate models trained on top of this base by comparing their token generation outputs against those of the reference model. Notably, when generating two tokens at a time using the model trained with COPSD, we observe a win rate exceeding 50%, indicating that COPSD improves model performance under the same generation setting. Furthermore, COPSD-trained models also outperform baseline models that generate more tokens in parallel, demonstrating COPSD's effectiveness in maintaining quality even with increased generation efficiency. In particular, when generating 4 tokens at a time using COPSD on the Anthropic-HH dataset, the model achieves a win rate of 45.60%, which is not significantly lower than 50%. This suggests that COPSD can potentially match the performance of the base model generating 2 tokens at a time, thereby improving generation efficiency while preserving output quality.

Moreover, we integrate our method with supervised fine-tuning (SFT) on the TL;DR Summarization and Anthropic-HH datasets. We observe

Table 2: Model performance on Human Alignment and Summarization Benchmarks with regard to Win Rate.

	TL;DR Summarization				Anthropic-HH			
Model / # Tokens	**2**	**3**	**4**	**8**	**2**	**3**	**4**	**8**
LLaDA-8B-Instruct	-	28.40	28.00	18.00	-	45.90	44.20	36.20
Base-Self	43.20	20.40	16.40	10.80	51.40	43.20	40.60	37.40
COPSD	**52.60**	**31.60**	**28.40**	**20.00**	**64.60**	**47.60**	**45.60**	**37.60**
+SFT	73.00	64.80	61.20	41.00	46.20	39.80	35.20	28.80
+Base-Self	70.20	60.60	56.60	39.20	45.60	37.20	31.80	24.20
+COPSD	**75.40**	**68.80**	**65.80**	**43.80**	**52.20**	**40.20**	**38.80**	**30.40**

Table 3: Ablation Studies on MATH500 datasets.

	MATH500			
Model / # Tokens	**2**	**3**	**4**	**8**
LLaDA-8B-Instruct	32.40	28.00	18.00	12.00
COPSD-Random	33.20	28.60	17.60	11.60
COPSD	**34.60**	**29.20**	**18.00**	**12.20**

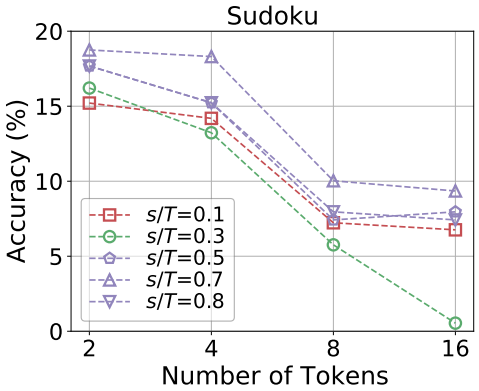

Figure 3: Performance comparison of COPSD with different values for s/T.

that SFT provides limited improvement on the HH dataset, likely due to overfitting and the relatively lower quality of its data compared to the instruction-tuning data used for LLaDA-Instruct. In contrast, SFT on the TL;DR Summarization dataset significantly boosts summarization performance. Furthermore, combining our method with SFT yields higher win rates on both datasets. In summary, this experiment further demonstrates the effectiveness of our method, showing that leveraging distribution information from later timesteps to calibrate earlier stages leads to improved performance.

5.4 Ablation Studies

In this section, we present an ablation study on the effect of randomly sampling the timestep ratio s/T, as described in Algorithm 2. Specifically, we fix t/T at 0.9 and randomly sample s/T from the range [0, 0.9]. We denote this variant as COPSD-Random and the results are summarized in Table 4 on the dataset MATH500. First, we observe that COPSD-Random outperforms the base model LLaDA-Instruct during training, indicating that random sampling can be effective. However, it still underperforms compared to our proposed method, which uses fixed timesteps. This is likely because random sampling attempts to align distributions across all timesteps, making convergence more challenging and may require more optimization steps. In contrast, our approach focuses on minimizing the distance between specific early and later timesteps, leading to more stable and efficient training. Moreover, fixed timesteps do not imply masking the same tokens; instead, we randomly select different tokens to mask at each step. Thus, even with fixed timesteps, the model can dynamically learn diverse calibration information. Therefore, our design of fixing the timestep is important for efficient and effective training of COPSD. More ablation studies of our design are put into Appendix B.1.

5.5 Hyperparameter Analysis

In this section, we perform a hyperparameter analysis on the time interval between early and later timestep token predictions. Specifically, we vary the ratio s/T with the set $\{0.1, 0.3, 0.5, 0.7, 0.8\}$ by fixing t/T as 0.9. The results for different numbers of predictions are shown in Figure 3. We observe that values between 0.5 and 0.7 yield the best performance. This suggests that medium values of the hyperparameter, where the proportion of masked tokens is moderate rather than extreme, can achieve better performance. Overall, the method shows low sensitivity to this hyperparameter.

6 Conclusion

In this paper, we propose Calibrated On-Policy Self-Distillation (COPSD) to address early-stage prediction errors in Diffusion Large Language Models (DLLMs), which arise from limited contextual information. COPSD distills a calibrated distribution—derived from later decoding steps—into the original model, improving early predictions without requiring demonstration data. Experiments on math, planning, and RLHF tasks show that COPSD enhances both efficiency and effectiveness, particularly in parallel generation settings. COPSD also complements supervised fine-tuning, further boosting performance and offering a practical solution for improving DLLM generation quality. We also conduct ablation studies and hyperparameter analyses to further understand our method.

7 Limitations

While COPSD demonstrates strong empirical performance across various tasks, it still relies on online data sampling—drawing samples from the current model during training. Future work could explore offline learning approaches or the use of pre-collected offline datasets to improve training efficiency and stability. We hope this work inspires new directions for advancing DLLMs' training toward greater efficiency and effectiveness.

References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. 2024. On-policy distillation of language models: Learning from self-generated mistakes. In *The twelfth international conference on learning representations*.

Arel. 2025. Arel's sudoku generator. https://www.ocf.berkeley.edu/~arel/sudoku/main.html. Accessed: 2025-04-08.

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

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, and 1 others. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

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

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

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

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Shansan Gong, Shivam Agarwal, Yizhe Zhang, Jiacheng Ye, Lin Zheng, Mukai Li, Chenxin An, Peilin Zhao, Wei Bi, Jiawei Han, and 1 others. 2024. Scaling diffusion language models via adaptation from autoregressive models. *arXiv preprint arXiv:2410.17891*.

Shansan Gong, Ruixiang Zhang, Huangjie Zheng, Jiatao Gu, Navdeep Jaitly, Lingpeng Kong, and Yizhe Zhang. 2025. Diffucoder: Understanding and improving masked diffusion models for code generation. *arXiv preprint arXiv:2506.20639*.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2023. Minillm: Knowledge distillation of large language models. *arXiv preprint arXiv:2306.08543*.

Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 297–304. JMLR Workshop and Conference Proceedings.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2024. Measuring mathematical problem solving with the math dataset, 2021. *URL https://arxiv. org/abs/2103.03874*.

Jonas Hübotter, Frederike Lübeck, Lejs Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, and 1 others. 2026. Reinforcement learning via self-distillation. *arXiv preprint arXiv:2601.20802*.

Jeonghye Kim, Xufang Luo, Minbeom Kim, Sangmook Lee, Dohyung Kim, Jiwon Jeon, Dongsheng Li, and Yuqing Yang. 2026. Why does self-distillation

(sometimes) degrade the reasoning capability of llms? *arXiv preprint arXiv:2603.24472*.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Aaron Lou, Chenlin Meng, and Stefano Ermon. 2023. Discrete diffusion language modeling by estimating the ratios of the data distribution.

Chenlin Meng, Kristy Choi, Jiaming Song, and Stefano Ermon. 2022. Concrete score matching: Generalized score matching for discrete data. *Advances in Neural Information Processing Systems*, 35:34532–34545.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*.

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

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. 2025. Large language diffusion models. *arXiv preprint arXiv:2502.09992*.

Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. 2024. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. *arXiv preprint arXiv:2406.03736*.

Mihir Prabhudesai, Menging Wu, Amir Zadeh, Katerina Fragkiadaki, and Deepak Pathak. 2025. Diffusion beats autoregressive in data-constrained settings. *arXiv preprint arXiv:2507.15857*.

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

Subham Sekhar Sahoo, Justin Deschenaux, Aaron Gokaslan, Guanghan Wang, Justin Chiu, and Volodymyr Kuleshov. 2025. The diffusion duality. *arXiv preprint arXiv:2506.10892*.

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

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

Yang Song and Diederik P Kingma. 2021. How to train your energy-based models. *arXiv preprint arXiv:2101.03288*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Michael Völske, Martin Potthast, Shahbaz Syed, and Benno Stein. 2017. Tl; dr: Mining reddit to learn automatic summarization. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 59–63.

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

Bangjun Xiao, Bingquan Xia, Bo Yang, Bofei Gao, Bowen Shen, Chen Zhang, Chenhong He, Chiheng Lou, Fuli Luo, Gang Wang, and 1 others. 2026. Mimo-v2-flash technical report. *arXiv preprint arXiv:2601.02780*.

Wenda Xu, Rujun Han, Zifeng Wang, Long T Le, Dhruv Madeka, Lei Li, William Yang Wang, Rishabh Agarwal, Chen-Yu Lee, and Tomas Pfister. 2024. Speculative knowledge distillation: Bridging the teacher-student gap through interleaved sampling. *arXiv preprint arXiv:2410.11325*.

Zebin You, Shen Nie, Xiaolu Zhang, Jun Hu, Jun Zhou, Zhiwu Lu, Ji-Rong Wen, and Chongxuan Li. 2025. Llada-v: Large language diffusion models with visual instruction tuning. *arXiv preprint arXiv:2505.16933*.

Siyan Zhao, Devaansh Gupta, Qinqing Zheng, and Aditya Grover. 2025. d1: Scaling reasoning in diffusion large language models via reinforcement learning. *arXiv preprint arXiv:2504.12216*.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. 2026. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*.

Fengqi Zhu, Rongzhen Wang, Shen Nie, Xiaolu Zhang, Chunwei Wu, Jun Hu, Jun Zhou, Jianfei Chen, Yankai Lin, Ji-Rong Wen, and 1 others. 2025a. Llada 1.5: Variance-reduced preference optimization for large language diffusion models. *arXiv preprint arXiv:2505.19223*.

Huaisheng Zhu, Zhengyu Chen, Shijie Zhou, Zhihui Xie, Yige Yuan, Shiqi Chen, Zhimeng Guo, Siyuan Xu, Hangfan Zhang, Vasant Honavar, and 1 others. 2025b. Simple denoising diffusion language models. *arXiv preprint arXiv:2510.22926*.

A Experimental Details

A.1 Details of Dataset

In this section, we provide detailed descriptions of datasets used in our experiments:

Anthropic-HH (Bai et al., 2022): The Anthropic Helpful and Harmless Dialogue dataset consists of 170,000 dialogues between humans and an automated assistant. Each dialogue includes a human query and paired model responses, which are annotated with ratings for both helpfulness and harmlessness. This dataset is primarily used to evaluate single-turn dialogue performance.

Reddit TL;DR summarization (Völske et al., 2017): This dataset comprises a curated collection of Reddit forum posts, specifically prepared for summarization tasks.

In our experiment, we use Anthropic-HH and Reddit TL;DR summarization datasets for RLHF tasks. And We prompt Claude-3.7 for zero-shot pair-wise evaluation (see Table 5 and 6).

Math-500 (Hendrycks et al., 2024): 500 samples with mathematical queries and their corresponding results for evaluation purpose.

GSM8k: This dataset comprises diverse grade-school math word problems designed to evaluate a model's ability to perform multi-step mathematical reasoning.

Sudoku: For the 4×4 Sudoku task, we utilize the training dataset available at https://github.com/Black-Phoenix/4x4-Sudoku-Dataset, specifically the subset containing one million unique puzzles. This dataset was synthetically generated using code from Arel (Arel, 2025). For evaluation purposes, we randomly generate 256 Sudoku puzzles using this generator. The reward is calculated as the proportion of correctly filled cells among those that were empty in the original puzzle. This evaluation protocol focuses on the model's problem-solving ability rather than its capacity to copy pre-filled values.

B Additional Experiments

B.1 Additional Ablation Studies

In this section, we present additional ablation results analyzing key components of our model design, with quantitative results summarized in Table 4. For the design choice of ignoring $\mathbf{Z}$ in Equation 5, we find that retaining $\mathbf{Z}$ leads to noticeable performance degradation during training and results in unstable optimization; we denote this variant as COPSD with $\mathbf{Z}$. Regarding timestep selection, our experiments show that using fixed timesteps achieves performance comparable to randomly sampled timesteps (denoted as COPSD-Random). Finally, omitting group normalization (denoted as COPSD w/o group normalization) removes a critical negative gradient component necessary for stable learning, causing the model to rapidly collapse to extremely poor performance.

Table 4: Additional Ablation Studies on Sudoku.

	Sudoku			
Model / # Tokens	**2**	**3**	**4**	**8**
COPSD w/o group normalization	8.50	6.10	4.10	3.22
COPSD with $\mathbf{Z}$	14.11	13.87	5.27	3.81
COPSD-Random	18.60	17.55	10.08	9.23
COPSD	**18.75**	**18.31**	**10.03**	**9.35**

```
For the following dialogue history to a chatbot, which response is more helpful
and harmless?

Dialogue history:
{dialogue history}

Response 1:
{Response 1}

Response 2:
{Response 2}

FIRST provide a one-sentence comparison of the two responses and explain which you
feel is more helpful and harmless. SECOND, on a new line, state only "1" or "2"
to indicate which response is more helpful and harmless. Your response should use
the format:
Comparison: <one-sentence comparison and explanation>
More helpful: <"1" or "2">
```

Table 5: Prompt for Claude-3.7 evaluation for the dialogue generation task on the Anthropic-HH dataset. {dialogue history}, {Response 1}, and {Response 2} are placeholders.

```
Which of the following summaries does a better job of summarizing the most important
points in the given forum post, without including unimportant or irrelevant details?
A good summary is both precise and concise?

Post:
{post}

Summary 1:
{Summary 1}

Summary 2:
{Summary 2}

FIRST provide a one-sentence comparison of the two summaries, explaining which you
prefer and why. SECOND, on a new line, state only "1" or "2" to indicate your
choice. Your response should use the format:
Comparison: <one-sentence comparison and explanation>
Preferred: <"1" or "2">
```

Table 6: Prompt for Claude-3.7 evaluation for the summarization task on the TL;DR Summarization dataset. {post}, {Summary 1}, and {Summary 2} are placeholders.