SelFusion: Self-distillation for Diffusion Language Models

Hyeong Soo Lim[1,*] **Jin Young Kim**[1,*] **Eun Seo Seo**[1] **Min Ho Jang**[1] **Ji Won Yoon**[1,†]
[1]Department of Artificial Intelligence, Chung-Ang University
{andrew1001, wlsdud338, jeo0534, sunbi8534, jiwonyoon}@cau.ac.kr
[*]Equal contribution [†]Corresponding author

Abstract

Diffusion language models (DLMs) alleviate the inherent latency bottleneck of autoregressive (AR) large language models (LLMs), but their degraded generation quality limits practical applicability. Although knowledge distillation (KD) can be a promising direction for improving performance, we empirically find that naively applying conventional KD yields only marginal gains, or even degrades generation quality. Based on these observations, we propose a novel self-distillation framework for DLMs, namely SelFusion. To enable effective KD without an external teacher model, SelFusion performs two forward passes with different masking levels, defining the hard mode with a larger masking probability and the easy mode with a smaller masking probability. However, the easy mode is not always more accurate than the hard mode and can be overconfident on incorrect tokens. Thus, we introduce bidirectional KD between the two modes, which can dynamically determine the distillation direction based on token-level correctness. Experimental results on instruction-following tasks show that the proposed self-distillation substantially outperforms other KD methods with external LLM and DLM teachers. In many configurations, the student trained with SelFusion even surpasses the performance of the LLM teacher, providing a practical path toward improving DLM generation quality. Source code can be found at https://github.com/scai-research/SelFusion_official

1 Introduction

Recently, diffusion language models (DLMs) have emerged as a compelling alternative autoregressive (AR) large language models (LLMs). By leveraging parallel decoding mechanisms, DLMs offer faster inference capabilities, making them highly suitable for real-time applications. LLaDA (Nie et al., 2025b) and SMDM(Nie et al., 2025a) have demonstrated notable inference speedups over AR

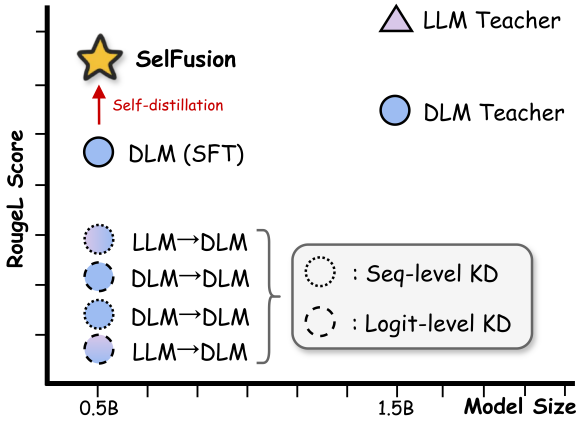

Figure 1: Rouge-L scores on the Dolly dataset. Existing KD methods for DLMs often underperform the SFT baseline, regardless of whether the teacher is DLM or LLM. In contrast, SelFusion significantly outperforms all baselines at the same model size.

counterparts. However, DLMs typically underperform LLMs in terms of generation quality due to their non-autogresive (NAR) nature (Nie et al., 2025b,a; Sahoo et al., 2024).

To bridge this performance gap, knowledge distillation (KD) (Hinton et al., 2015) can be a promising direction, enabling a student to mimic the behaviors of a strong teacher. In the context of LLMs, KD has been extensively studied and is typically categorized into two strategies. First, logit-level KD performs distribution matching between teacher and student, which is the most common approach (Chen et al., 2024; Gu et al., 2024a). Second, sequence-level KD trains the student on teacher-generated text, reamaining useful when teacher distributions are inaccessible (Kim and Rush, 2016). While these strategies have proven effective in improving LLMs, their extension to the DLMs remains largely underexplored.

To examine the applicability of KD to DLMs, we distill DLM students from both LLM and DLM teachers. Surprisingly, as shown in Figure 1, dis-

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 22077–22089

tilled students achieve only marginal gains, or even exhibit performance degradation. In the case of LLM-to-DLM distillation, the distribution mismatch between the AR teacher and the NAR student limits the effectiveness of logit-level knowledge transfer, which will be further discussed in Section 3.1. Sequence-level KD relies solely on teacher-generated outputs and thus provides improvements over logit-level supervision, but remains limited in terms of performance gains. Moreover, DLM-to-DLM KD scenarios remain suboptimal, largely due to the limited generation quality of the DLM teacher.

Motivated by these observations, we propose **SelFusion**, a novel self-distillation framework for DLMs. Specifically, SelFusion leverages the noising process of DLMs to enable two forward modes within a single model, namely the easy mode and the hard mode. The input to the easy mode has a lower masking ratio than that of the hard mode. Since fewer tokens are masked, the easy mode is expected to yield more accurate predictions and provide more beneficial knowledge for KD. However, it is not always more accurate than the hard mode and can also be overconfident on incorrect tokens. Thus, we introduce bidirectional KD between the easy and hard modes, dynamically determining the distillation direction by evaluationg token-level correctness.

We evaluate SelFusion on multiple instruction-following benchmarks against existing KD methods. The proposed self-distillation consistently outperforms conventional KD baselines that depend on external teacher models across all configurations. More surprisingly, SelFusion even surpasses the teacher models, including both DLMs and LLMs. These results suggest that self-distillation with two modes effectively transfers knowledge within a single model.

2 Related Work

2.1 Diffusion Language Models

DLMs for text generation can be categorized into continuous and discrete methods. Continuous methods embed tokens into continuous space, while discrete methods operate directly on token space (Gulrajani and Hashimoto, 2023). Recently, masked diffusion has been predominantly adopted among discrete approaches, where the forward process progressively masks tokens and the reverse process learns to predict them (Sahoo et al., 2024; Lou et al., 2024; Nie et al., 2025a,b). Generally, increasing the number of denoising steps improves generation quality. LLaDA scaled masked diffusion to 8B parameters, demonstrating the potential of DLMs for fast inference through parallel generation (Nie et al., 2025b). Despite these advances, DLMs still lag behind AR models in generation quality. For instance, recent DLMs underperform AR baselines by 10-32% in perplexity (Nie et al., 2025b). This gap highlights the need for further improvements in DLM generation quality.

2.2 Knowledge Distillation for Language Models

KD (Romero et al., 2015; Shridhar et al., 2023; Hsieh et al., 2023; Li et al., 2024; Jung et al., 2025) is a promising approach to improve model performance by transferring knowledge from a teacher model. Early work in AR models proposed logit-level KD that matches output logits (Hinton et al., 2015), followed by sequence-level KD that trains on teacher-generated sequences (Kim and Rush, 2016). Recent advances have improved distillation for generative models. MiniLLM (Gu et al., 2024a) addressed the limitations of forward KL divergence by proposing reverse KL divergence with on-policy optimization for instruction-following tasks, while GKD (Agarwal et al., 2024) explored on-policy distillation using student-generated samples. Self-distillation methods have also shown promise by training models to match their own predictions from different configurations (Hahn and Choi, 2019; Yoon et al., 2023; Yang et al., 2024). However, KD for DLMs to improve generation quality remains largely unexplored. Furthermore, existing work on DLM distillation has primarily focused on the pretraining phase, leaving post-training distillation scenarios unaddressed.

3 Methodology

3.1 Motivation

LLM-to-DLM Distillation. As aforementioned, logit-level distillation from an AR teacher is often ineffective due to distribution mismatch. We empirically observe that AR models assign near 100% probabilty so to the top-1 token, whereas DLMs exhibit approximately 60% probability at a 50% masking ratio. This gap hinders effective knowledge transfer, as the student struggles to match the teacher's spiky predictions. We also provide a detailed analysis of this mismatch in Section 4.3.5.

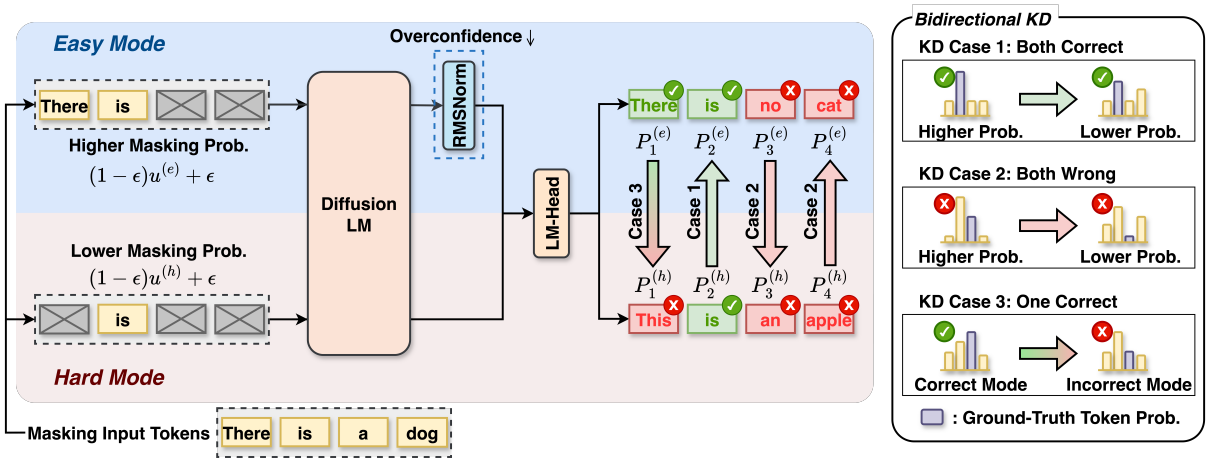

Figure 2: Overall architecture of SelFusion. The framework comprises easy and hard modes that generate tokens under more and less masked context, respectively. Bidirectional KD determines the distillation direction based on token-level correctness.

DLM-to-DLM Distillation. In the DLM-to-DLM setting, distillation is limited by the absence of sufficiently high-quality DLM teachers. Even when DLM teachers are available, their generation quality remains substantially lower than that of AR models. As a result, logit-level KD yields only marginal gains relative to its increased training cost. Sequence-level KD also shows limited effectiveness, even when the teacher generates outputs with more denoising steps.

3.2 SelFusion

Based on our findings that DLMs lack an effective teacher for KD, we propose a novel self-distillation, namely SelFusion. The overall process of SelFusion is illustrated in Figure 2.

Two Modes with Different Masking. The key idea is to leverage the noising process of DLMs to construct 'easy mode' and 'hard mode' within the same model. Specifically, while the hard mode follows the original random masking scheme, the easy mode is designed to use a lower masking ratio to expose more context, resulting in relatively higher masking ratios for the hard mode. As a result, the easy mode is expected to yield more accurate predictions. For example, in our experiments, the easy mode assigns approximately 10% higher probability to the correct token than the hard mode throughout training. This behavior is consistent with prior KD studies that emphasize student-friendly teachers, which maintain output distributions close to the student (Gu et al., 2024a; Kim et al., 2024; Lee et al., 2024). The easy mode tends to serve as the teacher for the hard mode, as it applies less masking and can thus provide relatively more accurate knowledge. Further details are provided in Section 4.3.3.

Bidirectional KD. However, the easy mode with lower masking does not always guarantee correct predictions, which motivates us to adaptively determine the distillation target. Therefore, we additionally present bidirectional KD, where the distillation direction for each token is determined based on correctness and confidence, considering three cases:

- Both correct: When both modes predict correctly, the mode assigning higher probability to the predicted token serves as the distillation target.
- Both wrong: When both modes predict incorrectly, the mode assigning higher probability to the ground-truth token serves as the distillation target.
- One correct: When only one mode predicts correctly, that mode serves as the distillation target.

Formally, the token-level distillation direction $\mathcal{D}_t$ is defined as follows:

$$\mathcal{D}_t = \begin{cases} e \to h, & \text{if } c_e > c_h, \\ h \to e, & \text{if } c_h > c_e, \\ \arg\max_{m \in \{h,e\}} p_m(y_e \mid x), & \text{if } c_h = c_e. \end{cases} \tag{1}$$

where c_h and c_e are binary indicators of correctness for the hard and easy mode predictions, respectively. Distillation therefore follows the more reliable prediction at the token level.

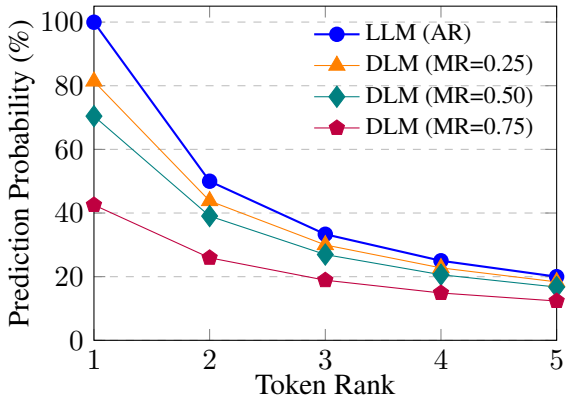

Figure 3: Comparison of average prediction probabilities for masked tokens in the Dolly dataset for LLM and DLM. For a fair comparison, we compute token probabilities using the same token prediction procedure as in training. Unlike AR models, DLMs exhibit flatter distributions, especially at higher MR.

RMSNorm-based Logit Calibration. Since the easy mode has access to more context during token generation, it tends to produce overconfident predictions. Figure 3 shows that lower masking ratios lead to more peaked distributions. More importantly, this overconfidence occurs not only on correct tokens but also on incorrect ones. Given that both modes predict incorrectly in approximately 40% of cases, such overconfidence from the easy mode can hinder effective distillation. To address this, we apply RMSNorm-based logit calibration. Specifically, we insert RMSNorm between the final hidden state and the LM head of the easy mode. This selectively suppresses overconfidence, preserving confidence for correct predictions while substantially reducing it for incorrect ones. With only 1,280 parameters, RMSNorm alleviates overconfident predictions, enabling more effective knowledge transfer. Further analysis of RMSNorm is provided in Section 4.3.4.

3.3 Objective Function for SelFusion

SelFusion jointly optimizes the hard mode diffusion loss, the easy mode diffusion loss, and the token-wise bidirectional distillation loss. Let x be an input sequence and y_i the ground-truth token at position i.

Masking Process and Notation. For each mode $m \in \{e, h\}$, we sample a noise level $u^{(m)} \in (0, 1)$ and define the per-position masking probability as follows:

$$q_i^{(m)} = (1 - \epsilon)\, u^{(m)} + \epsilon, \tag{2}$$

where ϵ is a small constant. We then sample a binary mask variable $z_i^{(m)} \sim \text{Bernoulli}(q_i^{(m)})$ independently for each position i. If $z_i^{(m)} = 1$, the token at position i is replaced by the special [MASK] token; otherwise it remains visible. We denote by $\mathcal{M}_m = \{i \mid z_i^{(m)} = 1\}$ the set of masked positions in mode m. The model output distribution at position i in mode m is denoted by $P_i^{(m)}(\cdot)$.

Diffusion Losses for Easy and Hard Mode. We compute the diffusion token-prediction losses over the masked positions, reweighted by the corresponding masking probabilities, which are defined as

$$\mathcal{L}_{\text{diff}}^{(h)} = \frac{1}{|\mathcal{M}_h|} \sum_{i \in \mathcal{M}_h} \frac{-\log P_i^{(h)}(y_i)}{q_i^{(h)}}, \tag{3}$$

$$\mathcal{L}_{\text{diff}}^{(e)} = \frac{1}{|\mathcal{M}_e|} \sum_{i \in \mathcal{M}_e} \frac{-\log P_i^{(e)}(y_i)}{q_i^{(e)}}. \tag{4}$$

Bidirectional KD Loss. Let $\mathcal{D}$ be the set of distillation positions. For each $i \in \mathcal{D}$, the distillation direction $\mathcal{D}_i \in \{e \to h,\ h \to e\}$ is determined by the rule defined in Eq. (1). With temperature T, we define the temperature-scaled distributions from the logits $z_i^{(m)}$ as follows:

$$P_{i,T}^{(m)}(\cdot) = \text{softmax}\left(z_i^{(m)}/T\right), \tag{5}$$

where $\mathcal{V}$ denotes the vocabulary (with size $|\mathcal{V}|$), where $z_i^{(m)} \in \mathbb{R}^{|\mathcal{V}|}$ denotes the logits at position i under mode $m \in \{h, e\}$. We use the Kullback-Leibler (KL) divergence to measure the discrepancy between two output distributions. The bidirectional KD loss is given by

$$\mathcal{L}_{\text{bkd}} = \frac{T^2}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} \text{KL}(P_{i,T}^{(j)} \| P_{i,T}^{(k)}), \tag{6}$$

where $(j, k) \in \{(e, h), (h, e)\}$ depending on the direction $\mathcal{D}_i$.

Total Objective. The final training objective can be calculated as

$$\mathcal{L}_{\text{SelFusion}} = \mathcal{L}_{\text{diff}}^{(h)} + \mathcal{L}_{\text{diff}}^{(e)} + \mathcal{L}_{\text{bkd}}. \tag{7}$$

Since both modes share the same parameters, the combined loss updates both modes simultaneously in a single backward pass.

Method	Model	Size	Dolly	Self-inst	Vicuna	Sinst	Uinst
SFT	LLM_{tea}	1476M	24.4765	11.0382	14.9436	23.2118	27.1273
	DLM_{tea} (8 Steps)		17.2631	9.7284	12.4577	22.6308	21.9190
	DLM_{tea} (16 Steps)		18.6108	10.5172	14.7385	24.3427	23.9995
	LLM_{stu}	472M	25.5660	11.1115	14.6017	22.4033	25.2105
	DLM_{stu} (8 Steps)		18.7688	10.6877	15.0473	24.7139	23.843
	DLM_{stu} (16 Steps)		19.5204	11.6416	16.3351	25.7185	25.5245
Method	**KD**	**Steps**	**Dolly**	**Self-inst**	**Vicuna**	**Sinst**	**Uinst**
Logit-level KD	$\text{LLM}_{tea} \rightarrow \text{DLM}_{stu}$	8	15.1075	8.7333	14.3417	16.5709	17.8998
	$\text{DLM}_{tea} \rightarrow \text{DLM}_{stu}$		17.2276	9.8326	14.4729	20.4459	19.8666
Seq-level KD	$\text{LLM}_{tea} \rightarrow \text{DLM}_{stu}$		18.8576	10.0266	14.3811	22.4884	22.6125
	$\text{DLM}_{tea} \rightarrow \text{DLM}_{stu}$		14.6822	8.3684	14.6454	16.2587	17.1783
Ours, SelFusion	$\text{DLM}_{stu} \leftrightarrow \text{DLM}_{stu}$		**21.3926**	**12.0022**	**16.6586**	**26.3464**	**26.4189**
Logit-level KD	$\text{LLM}_{tea} \rightarrow \text{DLM}_{stu}$	16	15.3833	9.0501	15.4732	16.5188	17.9967
	$\text{DLM}_{tea} \rightarrow \text{DLM}_{stu}$		17.6549	10.1753	16.0814	21.4601	21.4262
Seq-level KD	$\text{LLM}_{tea} \rightarrow \text{DLM}_{stu}$		19.9478	11.0604	16.1340	23.7022	24.4449
	$\text{DLM}_{tea} \rightarrow \text{DLM}_{stu}$		15.5604	9.5634	16.4462	16.9650	18.4398
Ours, SelFusion	$\text{DLM}_{stu} \leftrightarrow \text{DLM}_{stu}$		**21.6317**	**12.8707**	**17.0788**	**27.0745**	**27.8595**

Table 1: Performance comparison on multiple evaluation datasets. The first block reports SFT results of the teacher and student baselines, where LLM_{tea} and DLM_{tea} denote the teacher models and LLM_{stu} and DLM_{stu} denote the student baseline models. The second block reports KD results, where the KD column indicates the distillation direction. Bold indicates the best result.

4 Experiments

4.1 Experimental Settings

Datasets. Following previous studies (Gu et al., 2024a; Kim et al., 2024), we evaluated on instruction-following tasks, where the model generates responses conditioned on instructions. We used Databricks-Dolly-15K (Conover et al., 2023) as our training dataset, with 12K samples for training and 500 samples for evaluation. We evaluated on five instruction-following benchmarks, including Dolly (500 samples), Self-Inst (252 samples) (Wang et al., 2023), Vicuna (80 samples) (Peng et al., 2023), and the $[11, +\infty)$ response-length subsets of S-NI (1,694 samples) (Wang et al., 2022) and UnNI (23,916 samples) (Honovich et al., 2023). The five benchmarks described above are the evaluation datasets reported in Table 1.

Models and Training Setup. All experiments were conducted using SMDM architectures with 472M and 1476M parameters (Nie et al., 2025a), with the number of training epochs fixed to 20. To identify the optimal configuration for each model and method, we explored various learning rates and epochs. The LLM teacher used in our experiments was a TinyLlama model with 1,476M parameters. We used the DLM and LLM teachers from (Nie et al., 2025a), where both models were trained under the same setup with matched training data, model size, and training epochs. Comprehensive details regarding the hyperparameter search space and final configurations are provided in the Appendix A. All models were evaluated using the checkpoint from the final training step. Experiments were executed on four NVIDIA H200 GPUs, each with 141GB of memory.

Evaluation Configurations. Following prior work on DLMs (Nie et al., 2025a), we fixed the classifier-free guidance (CFG) scale to 1.0 for all DLMs. For LLMs, the sampling temperature was set to 1.0 during evaluation. Generation quality was assessed using the ROUGE-L metric (Lin, 2004), which is widely adopted for evaluating instruction-following text generation. We evaluated each benchmark using three different random seeds and report the average across the three runs. Additional implementation and training details are provided in the Appendix A.

Method	KD	Student training	Teacher training	Total
SFT	—	2.8×10^2	—	2.8×10^2
Seq-level KD	$\text{LLM}_{\text{tea}} \rightarrow \text{DLM}_{\text{stu}}$	2.8×10^2	7.5×10^2	1.03×10^3
	$\text{DLM}_{\text{tea}} \rightarrow \text{DLM}_{\text{stu}}$	2.8×10^2	7.5×10^2	1.03×10^3
Logit-level KD	$\text{DLM}_{\text{tea}} \rightarrow \text{DLM}_{\text{stu}}$	5.4×10^2	7.5×10^2	1.29×10^3
	$\text{LLM}_{\text{tea}} \rightarrow \text{DLM}_{\text{stu}}$	5.4×10^2	7.5×10^2	1.29×10^3
Ours, SelFusion	$\text{DLM}_{\text{stu}} \leftrightarrow \text{DLM}_{\text{stu}}$	5.6×10^2	—	$\mathbf{5.6 \times 10^2}$

Table 2: Training cost analysis measured in TFLOPs. Although SelFusion requires higher per-iteration computation than SFT, it eliminates teacher training and thus reduces total training computation compared to KD methods that rely on a separately trained teacher. All methods are compared under the same training setup with 485 iterations.

4.2 Experimental Results

Firstly, we evaluated conventional logit-level and sequence-level KD in both the LLM-to-DLM and DLM-to-DLM settings, as shown in Table 1. For logit-level KD, we minimized the KL divergence between the teacher and student distributions. Since the student was a DLM, we applied the KD loss only to the masked tokens, following the DLM generation principle. In the case of sequence-level KD, we trained the student with teacher-generated outputs, as described in Section 2. This approach required the teacher that could generate high-quality target sequences to provide effective supervision (Kim and Rush, 2016). Prior work showed that DLMs' generation quality can be improved by increasing the number of diffusion steps (Deschenaux and Gulcehre, 2025; Nie et al., 2025a; Chen et al., 2025). Thus, in the DLM-to-DLM setting, we generated teacher pseudo-targets using 64-step inference and performed sequence-level KD on these sequences. We presented results for 8 and 16 diffusion steps in Table 1.

We began by evaluating LLM-to-DLM distillation with 8 diffusion steps. From the results, it is confirmed that logit-level KD with the LLM teacher led to substantial performance degradation. For example, on Dolly, the score dropped to 15.11, compared to 18.77 for the DLM SFT baseline. This trend was consistent across all benchmarks, indicating that direct logit matching from the LLM teacher to the DLM student was ineffective. Sequence-level KD also did not surpass the DLM SFT baseline on most benchmarks, with Dolly as the only exception. We next evaluated DLM-to-DLM distillation using the pretrained DLM teacher. Distillation did not improve over the SFT baseline. For example, on Uinst with 8 diffusion steps, the DLM SFT model achieved 23.84, whereas logit-level KD reached only 19.87. Sequence-level KD further exhibited performance drops across benchmarks. These results suggested that the DLM teacher's generation quality was insufficient to provide beneficial knowledge at either the sequence or logit level.

In contrast, the proposed self-distillation method achieved substantial performance gains without relying on any external teacher model. Table 1 shows that SelFusion outperformed the strongest competing baseline, LLM-to-DLM sequence-level KD, by 2 to 4 points, corresponding to an approximate 25% relative improvement. Compared with the DLM SFT baseline, it consistently improved performance by 1.5 to 3 points across all five benchmarks. Notably, SelFusion also surpassed the LLM teacher on multiple benchmarks. For example, SelFusion achieved 12.87 on Self-inst and 17.08 on Vicuna, exceeding the LLM teacher scores of 10.95 and 14.90, respectively. It also improved from 23.17 to 27.07 on Sinst and from 27.06 to 27.86 on Uinst. Overall, these results demonstrated that our self-distillation design provides a practical path for DLMs, requiring no external teacher and introducing only 1,280 additional parameters.

4.3 Analysis

4.3.1 Training Efficiency

We analyzed the training cost of SelFusion using TFLOPs. As shown in Table 2, SelFusion required about 2× more per-iteration computation than SFT, but substantially less total computation than KD methods with a separately trained teacher. By removing the teacher-training stage, SelFusion reduced the overall training cost by roughly 2× while maintaining competitive performance. All results were measured under the same training configura-

Method	Steps	Dolly	Self-inst	Vicuna	Sinst	Uinst
SelFusion	16	**21.6317**	**12.8707**	**17.0788**	**27.0745**	**27.8595**
w/o bidirectional KD (easy→hard)		13.6755 (-7.9562)	8.5558 (-4.3149)	10.6129 (-6.4659)	21.8953 (-5.1792)	20.9376 (-6.9219)
w/o bidirectional KD (hard→easy)		15.7107 (-5.9210)	9.6470 (-3.2237)	11.0940 (-5.9848)	23.9227 (-3.1518)	23.2298 (-4.6297)
w/o RMSNorm		17.3566 (-4.2751)	11.0430 (-1.8277)	16.4204 (-0.6584)	21.2283 (-5.8462)	21.9992 (-5.8603)

Table 3: Ablation results of SelFusion. We perform ablation studies by individually removing bidirectional KD and RMSNorm. For bidirectional KD, we further examine each unidirectional variant (easy→hard and hard→easy) to isolate the contribution of each direction. Across all benchmarks, removing either component leads to consistent performance degradation, and neither unidirectional variant matches the bidirectional setting

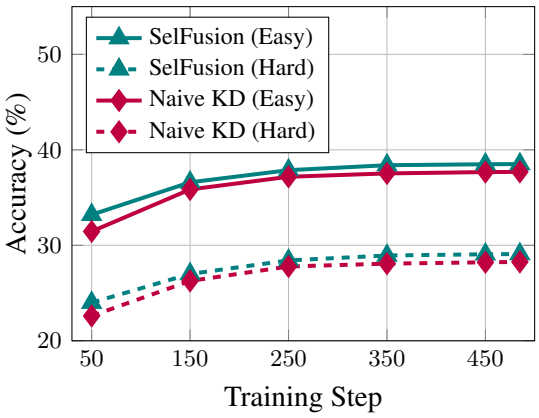

Figure 4: Training accuracy comparison between SelFusion and the naive KD baseline. Naive KD uses a fixed one-way distillation direction between the two modes, Easy→Hard. Accuracy is evaluated only on the masked positions for each of the Easy and Hard modes, which use different masking ratios.

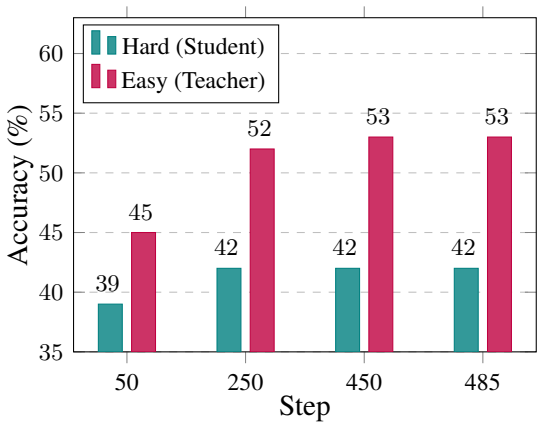

Figure 5: Step-wise comparison of the mean probability assigned to the ground truth tokens by the easy and hard modes during training.

tion with 485 iterations.

4.3.2 Effect of Bidirectional Distillation

We further analyzed the effect of distillation direction by comparing easy→hard, hard→easy, and bidirectional KD. Although the easy mode is expected to be a stronger teacher due to its richer visible context, Table 3 shows that hard→easy outperformed easy→hard on several benchmarks, while neither unidirectional direction matched bidirectional distillation. As illustrated in Figure 2, SelFusion dynamically switches the token-level distillation direction between the easy and hard modes based on accuracy and confidence, allowing each mode to guide the other when it produces more confident predictions. Figure 4 provides quantitative support for this behavior. Under identical settings, easy→hard distillation (denoted as Naive KD) resulted in an approximately 1% drop in masked token accuracy for both modes during training. Consistent with this, Table 3 shows that removing bidirectional KD causes substantial performance drops across all benchmarks, highlighting the importance of dynamic distillation target selection in self-distillation.

4.3.3 Token-level Probability Comparison of Easy and Hard Modes

To verify whether the easy mode assigns higher probability to ground-truth tokens than the hard mode, we analyze the model outputs under a controlled masking setup. Figure 5 presents the mean probability assigned to ground-truth tokens across training steps. Specifically, we evaluated step-wise checkpoints of SelFusion by fixing the hard mode masking ratio to 60% and using a reduced ratio

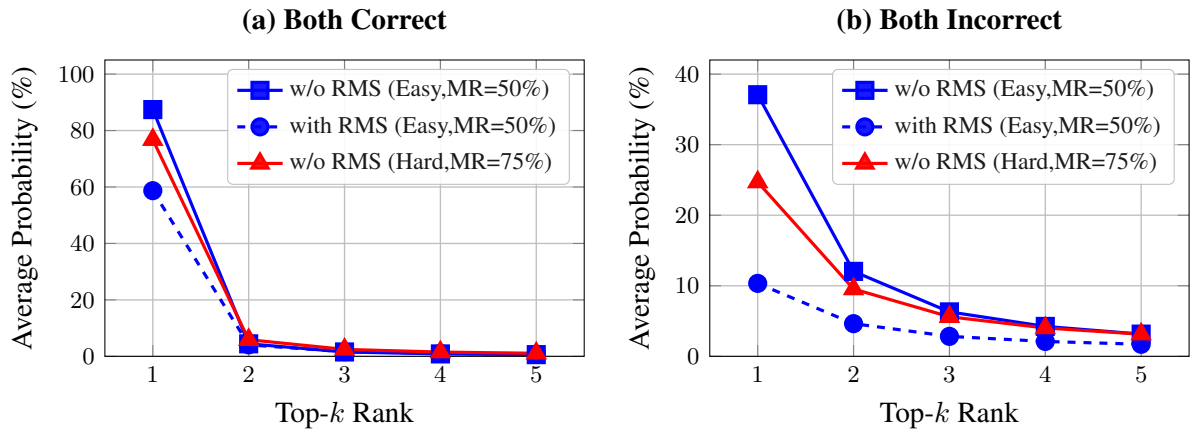

Figure 6: Analysis of RMSNorm effect on SelFusion. We analyze tokens that are masked in both modes. (a) shows the probability distribution when both modes are correct, while (b) shows the case when both modes are incorrect. Easy mode uses 50% masking ratio (MR), and hard mode uses 75% MR.

of 30% for the easy mode, where the easy mode mask was constructed as a subset of the hard mode mask. We then measured the mean probability assigned to ground-truth tokens over this shared masked subset. The easy mode consistently assigned higher probability to the correct token than the hard mode, with the gap increasing from approximately 6% to about 10% over training. This observation supported our design intuition of treating the easy mode as the teacher and the hard mode as the student. Importantly, the easy mode maintains an output distribution that is more closely aligned with the student model, thereby facilitating effective knowledge transfer in line with prior student-friendly KD principles (Gu et al., 2024b; Kim et al., 2024).

4.3.4 Logit Calibration by RMSNorm

We applied RMSNorm-based logit calibration to mitigate overconfident predictions from the easy mode. As shown in Figure 6, RMSNorm selectively calibrated confidence depending on prediction correctness. When both modes were correct, RMSNorm moderately reduced the top-1 probability from approximately around 90% to about 60%. More importantly, when both modes were incorrect, RMSNorm substantially suppressed the overconfident probability from around 40% to about 10%, approximately 75% reduction. This stronger calibration on incorrect predictions was crucial, as it prevented learning from unreliable signals. Table 3 further confirmed that removing RMSNorm resulted in consistent performance drops, indicating its importance in suppressing overly confident incorrect predictions during distillation.

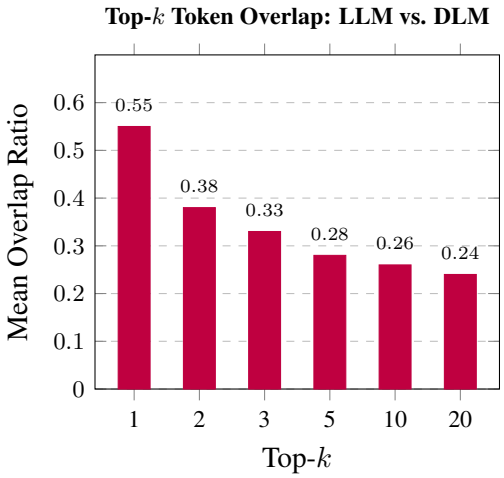

Figure 7: Mean top-k token overlap between LLM and DLM fine-tuned on the Dolly dataset. For each position, overlap is computed as $|S_k^{\text{LLM}} \cap S_k^{\text{DLM}}|/k$, where S_k denotes the set of top-k predicted tokens.

4.3.5 Distribution Mismatch between LLMs and DLMs

The primary challenge of LLM-to-DLM distillation stemmed from logit distribution mismatch caused by different generation mechanisms. We categorized this mismatch into two types: (1) top-k logit scale mismatch and (2) top-k token mismatch. As shown in Figure 3, LLMs and DLMs exhibited different probability scales: LLMs showed peaked distributions dominated by the top-1 token, whereas DLMs exhibited flatter distributions due to parallel generation. Figure 7 also showed limited top-k token overlap, with top-1 overlap at about 60% and decreasing as k increased. These mismatches hindered direct logit-level distillation from LLMs to

Method	KD	Steps	Dolly	Self-inst	Vicuna	Sinst	Uinst
Logit-level KD	$\text{LLM}_{tea} \rightarrow \text{DLM}_{stu}$	32	14.7424	8.9812	15.2372	15.6568	17.1595
	$\text{DLM}_{tea} \rightarrow \text{DLM}_{stu}$		15.9906	9.8436	17.2801	16.9256	18.6831
Seq-level KD	$\text{LLM}_{tea} \rightarrow \text{DLM}_{stu}$		20.1426	11.7443	16.6505	24.1393	25.1063
	$\text{DLM}_{tea} \rightarrow \text{DLM}_{stu}$		15.9906	9.8436	17.2801	16.9256	18.6831
Ours, SelFusion	$\text{DLM}_{stu} \leftrightarrow \text{DLM}_{stu}$		**21.7828**	**12.4917**	**17.1310**	**27.1185**	**28.1247**

Table 4: Ablation on larger diffusion steps. We evaluate DLMs with 32 diffusion steps to assess generalization beyond the main 8 and 16 step settings. Bold indicates the best result.

	Step	Latency (ms)	Speed-up
AR	–	2240	1.00×
DLM	1	61.4	36.5×
	2	109.7	20.4×
	4	199.1	11.3×
	8	378.4	5.9×
	16	745.4	3.0×
	32	1479.0	1.5×
	64	2952.2	0.76×

Table 5: Per-sample inference latency and speed-up for AR and DLM with varying diffusion steps, measured on the Dolly validation set.

DLMs, which SelFusion addressed via the inherent generation mechanism of DLMs.

4.3.6 Generalization on Larger Steps

We further evaluated its generalization to larger diffusion step settings. Beyond the standard 8 and 16 steps, we additionally evaluated 32-step inference. As shown in Table 4, SelFusion consistently outperformed baseline distillation methods under larger-step settings, suggesting that its gains were not specific to a particular step configuration.

4.4 Time Comparison of DLM and LLM

We compared the inference speed of DLMs and AR language models (LLMs). As shown in Table 5, DLMs achieved significantly lower inference latency than LLMs. This advantage stemmed from the parallel token generation of DLMs, whereas LLMs generated tokens sequentially. As a result, even with 32 diffusion steps, DLMs achieved approximately a 1.5× speedup over LLMs in practice.

5 Conclusions

In this paper, we propose SelFusion, a novel self-distillation framework that enables effective logit-level KD for DLMs. By leveraging different masking ratios with simultaneous forward passes, we decompose the model into two modes. This provides a more suitable distribution for learning, enabling effective training within a single model. Experimental results show that SelFusion outperforms conventional KD methods without relying on external teacher models.

Limitations

Despite SelFusion's consistent performance gains, several limitations remain. First, the current availability of DLM backbones is limited, constraining validation across a broader set of architectures. Second, our evaluation is limited to instruction tuning in English, leaving broader domains and multilingual settings for future work. Despite these limitations, SelFusion offers a practical advantage as a distillation framework that does not rely on external teacher models, including LLMs or DLMs.

Ethical Considerations

This work does not raise any ethical concerns.

Acknowledgments

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (RS-2025-02653113, High-Performance Research AI Computing Infrastructure Support at the 2 PFLOPS Scale). This work was also supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (RS-2021-II211341, Artificial Intelligence Graduate School Program (Chung-Ang University)), and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (RS-2025-00515722).

References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos, Matthieu Geist, and Olivier Bachem. 2024. On-policy distillation of language models: Learning from self-generated mistakes. In *Proc. NeurIPS*.

Hongzhan Chen, Ruijun Chen, Yuqi Yi, Xiaojun Quan, Chenliang Li, Ming Yan, and Ji Zhang. 2024. Knowledge distillation of black-box large language models. *Preprint*, arXiv:2401.07013.

Tianqi Chen, Shujian Zhang, and Mingyuan Zhou. 2025. Dlm-one: Diffusion language models for one-step sequence generation. *Preprint*, arXiv:2506.00290.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. 2023. Free dolly: Introducing the world's first truly open instruction-tuned llm.

Justin Deschenaux and Caglar Gulcehre. 2025. Beyond autoregression: Fast LLMs via self-distillation through time. In *proc. ICLR*.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. SAMSum corpus: A human-annotated dialogue dataset for abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 70–79, Hong Kong, China. Association for Computational Linguistics.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2024a. Minillm: Knowledge distillation of large language models. In *Proc. ICLR*.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2024b. Minillm: Knowledge distillation of large language models. In *Proc. ICLR*.

Ishaan Gulrajani and Tatsunori B. Hashimoto. 2023. Likelihood-based diffusion language models. In *Advances in Neural Information Processing Systems*, volume 36.

Sangchul Hahn and Heeyoul Choi. 2019. Self-knowledge distillation in natural language processing. In *Proc. RANLP*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *Preprint*, arXiv:1503.02531.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. 2023. Unnatural instructions: Tuning language models with (almost) no human labor. In *Proc. ACL*.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8003–8017, Toronto, Canada. Association for Computational Linguistics.

Seongryong Jung, Suwan Yoon, DongGeon Kim, and Hwanhee Lee. 2025. Todi: Token-wise distillation via fine-grained divergence control. *Preprint*, arXiv:2505.16297.

Gyeongman Kim, Doohyuk Jang, and Eunho Yang. 2024. Promptkd: Distilling student-friendly knowledge for generative language models via prompt tuning. *Preprint*, arXiv:2402.12842.

Yoon Kim and Alexander M. Rush. 2016. Sequence-level knowledge distillation. In *Proc. EMNLP*.

Hojae Lee, Junho Kim, and SangKeun Lee. 2024. Mentor-kd: Making small language models better multi-step reasoners. In *Proc. EMNLP*.

Zheng Li, Xiang Li, Xinyi Fu, Xin Zhang, Weiqiang Wang, Shuo Chen, and Jian Yang. 2024. Promptkd: Unsupervised prompt distillation for vision-language models. In *Proc. CVPR*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *proc. ACL(Workshop)*.

Aaron Lou, Chenlin Meng, and Stefano Ermon. 2024. Discrete diffusion modeling by estimating the ratios of the data distribution. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 32819–32848. PMLR.

Shen Nie, Fengqi Zhu, Chao Du, Tianyu Pang, Qian Liu, Guangtao Zeng, and Min Lin. 2025a. Scaling up masked diffusion models on text. *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. 2025b. Large language diffusion models. *Preprint*, arXiv:2502.09992.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. 2015. Fitnets: Hints for thin deep nets. In *Proc. ICLR*.

Subham Sekhar Sahoo, Marianne Arriola, Aaron Gokaslan, Edgar Mariano Marroquin, Alexander M Rush, Yair Schiff, Justin T Chiu, and Volodymyr Kuleshov. 2024. Simple and effective masked diffusion language models. In *proc. NeurIPS*.

Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. 2023. Distilling reasoning capabilities into smaller language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7059–7073, Toronto, Canada. Association for Computational Linguistics.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-Instruct: Aligning language models with self-generated instructions. In *Proc. ACL*.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Gary Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Maitreya Patel, and 21 others. 2022. Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks. In *Proc. EMNLP*.

Zhaorui Yang, Tianyu Pang, Haozhe Feng, Han Wang, Wei Chen, Minfeng Zhu, and Qian Liu. 2024. Self-distillation bridges distribution gap in language model fine-tuning. In *proc. ACL*.

Ji Won Yoon, Sunghwan Ahn, Hyeonseung Lee, Minchan Kim, Seok Min Kim, and Nam Soo Kim. 2023. EM-network: Oracle guided self-distillation for sequence learning. In *proc. ICML*.

A Appendix

A.1 Training Configuration

We present the detailed training configurations used in our experiments in Table 6. We determined the optimal learning rates through grid search, taking into account both model scale and architectural differences. For the 1.4B teacher models, including both AR and DLM architectures, we explored a wider range of learning rates, {1e-5, 5e-5, 1e-4, 2e-4}, due to their distinct architectural characteristics. For the 0.472B DLM target models, we conducted grid search over {5e-5, 1e-4, 2e-4}. Following the same procedure, we also selected the learning rate for SelFusion via grid search and used 5e-5 in the final configuration. The table also reports hyperparameters shared across all experimental settings. For evaluation, we used three random seeds, 10, 20, and 30, and report the average over these runs.

A.2 Prompt Formatting for Instruction Tuning

We standardized the instruction tuning data by converting each example into a Dolly style prompt template. Specifically, when an example contains an **input field**, we construct the prompt as follows:

Stage	Setting	Value
SFT	DLM (472M)	5×10^{-5}
	DLM (1.476B)	1×10^{-5}
	LLM (472M)	2×10^{-4}
	LLM (1.476B)	1×10^{-4}
KD	DLM→DLM (logit KD)	5×10^{-5}
	DLM→DLM (seq KD)	5×10^{-5}
	LLM→DLM (logit KD)	5×10^{-5}
	LLM→DLM (seq KD)	5×10^{-5}
Shared hyperparameters		**Value**
# devices		4
Global batch size		512
Max tokens		256
Epoch (final)		20
LR decay		enabled
Warmup ratio		0.05
Min LR		LR/10
Weight decay		0.1
Adam betas		(0.9, 0.95)
Grad clip		1.0
Seed		3407

Table 6: Selected hyperparameters from grid search and shared training settings.

```
Below is an instruction that describes a task,
paired with an input that provides further
context. Write a response that appropriately
completes the request.

### Instruction:
{instruction}

### Input:
{context}

### Response:
```

When the **input field is absent**, the following template is used:

```
Below is an instruction that describes a task.
Write a response that appropriately completes
the request.

### Instruction:
{instruction}

### Response:
```

All datasets used in our experiments are publicly available from the **MiniLLM** data release: https://github.com/microsoft/LMOps/tree/main/minillm

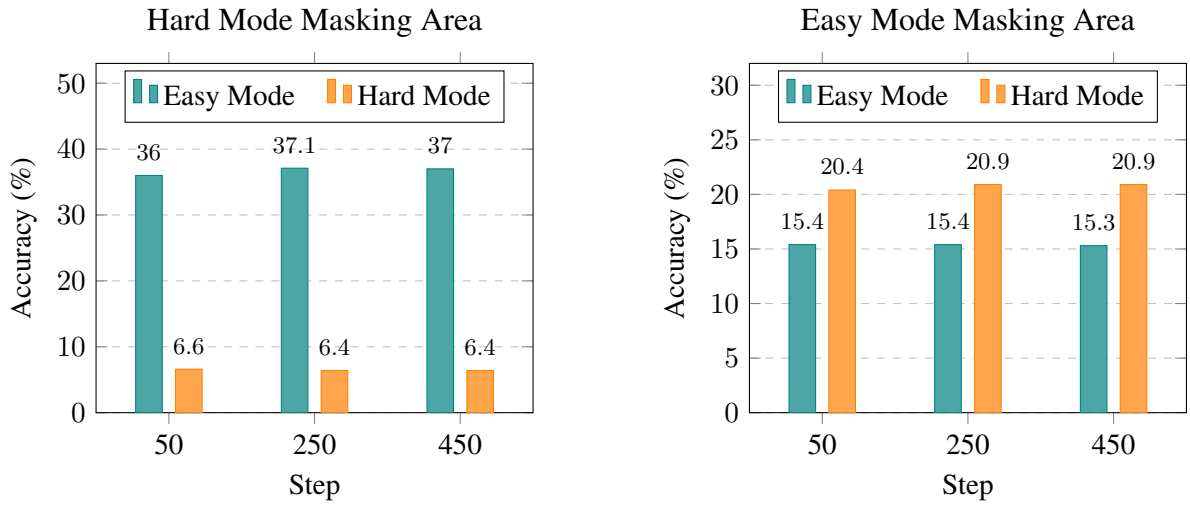

Figure 8: Masked-token prediction accuracy comparison between the easy and hard modes. The left plot evaluates both modes on hard mode masked positions, and the right plot evaluates both modes on easy mode masked positions. Note that a token masked in one mode may remain visible in the other mode. Thus, for a token masked in one mode, the counterpart mode may observe the ground truth token at that position.

A.3 Masking Strategy for Easy and Hard Modes

We specify the masking configuration for the easy and hard modes following the standard DLM noising procedure. Given an input sequence $\mathbf{x} \in \{0, \ldots, V-1\}^L$, we sample a noise level $t \sim \mathcal{U}(0,1)$ for each example and convert it into a token masking probability

$$p_{\text{mask}}(t) = (1-\epsilon)\, t + \epsilon, \tag{8}$$

where ϵ is a small constant to avoid degenerate masking. We then independently mask each position i with probability $p_{\text{mask}}(t)$ and replace masked tokens with a dedicated mask token (implemented by using the vocabulary index V):

$$\tilde{x}_i = \begin{cases} \texttt{[MASK]} & \text{with prob. } p_{\text{mask}}(t), \\ x_i & \text{otherwise.} \end{cases} \tag{9}$$

To construct paired easy and hard modes within a single training step, we first sample the hard mode noise level $t_{\text{hard}} \sim \mathcal{U}(0,1)$. We then sample the easy mode noise level conditioned on it as

$$t_{\text{easy}} \sim \mathcal{U}(0, t_{\text{hard}}), \tag{10}$$

which ensures $t_{\text{easy}} \leq t_{\text{hard}}$ and thus $p_{\text{mask}}(t_{\text{easy}}) \leq p_{\text{mask}}(t_{\text{hard}})$. Accordingly, the easy mode observes more visible context (lower masking), while the hard mode operates under reduced visibility (higher masking). Although the easy mode masking ratio is determined by conditioning on the hard mode noise level, the specific masked token positions are sampled independently for the two modes.

A.4 Token level Accuracy Comparison Details

To examine whether bidirectional KD is activated during training, we analyze the probability that only one of the two modes correctly predicts a masked token. As shown in Figure 8, we measure this probability on masked token positions for each mode. In the left plot, which evaluates hard mode masked tokens, we observe that only the hard mode predicts the correct token in approximately 6% of cases, whereas the easy mode alone is correct in about 37% of cases. In contrast, in the right plot corresponding to easy mode masked tokens, the hard mode correctly predicts the token in around 20% of cases, while the easy mode alone is correct in only about 15% of cases. These results indicate that even on its own masked positions, the easy mode is not always more accurate, and the hard mode can provide more accurate token predictions depending on the masking configuration. This complementary behavior explains why SelFusion benefits from bidirectional KD, as distillation can dynamically proceed from the mode with higher token accuracy at each masked position.

A.5 Generalization Across Tasks

To evaluate the generality of SelFusion beyond instruction-following, we further tested it on the summarization benchmark SAMSum (Gliwa et al., 2019). As shown in Table 7, SelFusion outperformed strong baselines, including SFT and existing KD methods, in both the 8-step and 16-step

Method	KD	8 Steps	16 Steps
SFT	—	27.9175	29.3172
Seq-level KD	$\text{LLM}_{tea} \rightarrow \text{DLM}_{stu}$	27.7835	29.2686
	$\text{DLM}_{tea} \rightarrow \text{DLM}_{stu}$	21.9711	23.4053
Logit-level KD	$\text{LLM}_{tea} \rightarrow \text{DLM}_{stu}$	26.3727	26.6891
	$\text{DLM}_{tea} \rightarrow \text{DLM}_{stu}$	28.9661	30.0167
Ours, SelFusion	$\text{DLM}_{stu} \leftrightarrow \text{DLM}_{stu}$	**29.3575**	**30.4928**

Table 7: Results on SAMSum measured by ROUGE-L across different diffusion steps. SelFusion consistently outperforms strong baselines, with larger gains under the more efficient 8-step setting. Bold indicates the best result.

settings. The improvement was more pronounced in the 8-step setting, indicating that SelFusion remained effective under tighter inference budgets.