Improving Text Style Transfer using Masked Diffusion Language Models with Inference-time Scaling

Tejomay Kishor Padole a,*, Suyash P. Awate and Pushpak Bhattacharyya

^aDept. of Computer Science and Engineering, Indian Institute of Technology, Bombay

Abstract. Masked diffusion language models (MDMs) have recently gained traction as a viable generative framework for natural language. This can be attributed to its scalability and ease of training compared to other diffusion model paradigms for discrete data, establishing itself as the state-of-the-art non-autoregressive generator for discrete data. Diffusion models, in general, have shown excellent ability to improve the generation quality by leveraging inferencetime scaling either by increasing the number of denoising steps or by using external verifiers on top of the outputs of each step to guide the generation. In this work, we propose a verifier-based inferencetime scaling method that aids in finding a better candidate generation during the denoising process of the MDM. Our experiments demonstrate the application of MDMs for standard text-style transfer tasks and establish MDMs as a better alternative to autoregressive language models. Additionally, we show that a simple soft-value-based verifier setup for MDMs using off-the-shelf pre-trained embedding models leads to significant gains in generation quality even when used on top of typical classifier-free guidance setups in the existing literature.

I Introduction

The current landscape of natural language generation is dominated by pre-trained autoregressive large language models (LLMs). The next-token prediction strategy these models employ has proven to be very effective for generative modeling on language data that is typically provided with large-scale corpora [4, 10]. However, sequential sampling also comes with limitations. Specifically, such a strategy suffers from "sampling drifts", meaning that the generation degrades as it progresses due to error accumulation [21]. Diffusion language modeling offers an alternative strategy by proposing non-autoregressive sampling of tokens with iterative denoising, potentially mitigating these downsides. Recently, the masked diffusion language modeling (MDM) paradigm has been proposed as a scalable approach that has been quite competitive with autoregressive models on the same parameter scale [30, 25, 26].

Apart from non-autoregressive generation, diffusion models have also demonstrated excellent abilities in guiding the generation towards specific outputs. Diffusion guidance techniques can be mainly classified as: (1) classifier guidance [8] where an external classifier is trained, which provides gradient-based guidance signals to the diffusion model during sampling, (2) classifier-free guidance (CFG) [12] where a diffusion model is trained jointly for conditional and unconditional generation which improves fidelity during sampling,

and (3) training-free guidance (TFG) [37] where off-the-shelf pretrained models are used directly to provide gradient-based guidance signals to a diffusion model. Apart from these, derivative-free guidance techniques have recently emerged as an alternative to classifier guidance and TFG, eliminating the gradient-computation step during sampling and easing the adaptation of off-the-shelf pre-trained models for guidance. Such techniques either follow a sequential Monte Carlo approach to improve the conditioning [9] or perform explicit reward maximization by setting the pre-trained model's likelihood/ classifier logits as rewards [16].

Guidance mechanisms in diffusion models are effective methods to scale the inference-time compute in diffusion models [23]. For instance, external models (either trained specifically for a task or used off-the-shelf) that offer guidance signals to diffusion models are seen as verifiers that consume extra compute. In return, they provide more control and potentially improve generation quality when combined with search algorithms. Existing work on diffusion models for discrete data has proposed methods to apply guidance mechanisms in discrete spaces [31, 27, 16]. However, their experiments are mostly restricted to discrete data modalities such as small molecules, DNA sequences, and protein sequences, while the language modality remains largely unexplored. The work of [15] showcases discrete diffusion guidance on language with sequential Monte Carlo but only applies it to attribute control for unconditional text generation. The work of [22] is conceptually closest to ours, which explores style transfer with seq-to-seq continuous diffusion [11], but they do not explore any guidance mechanisms or discuss inference-time scaling with external verifier-based search algorithms.

In this paper, we propose verifier-based inference-time scaling with masked diffusion language models (MDMs) for text-style transfer, one of the well-studied and important conditional language generation tasks. Specifically, we design a novel derivative-free guidance method with MDMs using pre-trained sentence-embedding models as external verifiers. We demonstrate the effectiveness of this setup using two standard text-style transfer tasks. To the best of our knowledge, we are the first to propose this setting for any conditional language generation task. Our contributions are

- 1. a novel derivative-free guidance method for MDMs using guidance signals from off-the-shelf pre-trained sentence embedding models to enhance conditional language generation. (Section 3.2).
- 2. demonstration (for the first time, to the best of our knowledge) of the utility of MDMs for text style transfer (with available parallel data for training) by performing seq-to-seq fine-tuning on pretrained MDMs (Section 3).

^{*} Corresponding Author. Email: tejomaypadole@cse.iitb.ac.in

¹ Code can be found here.

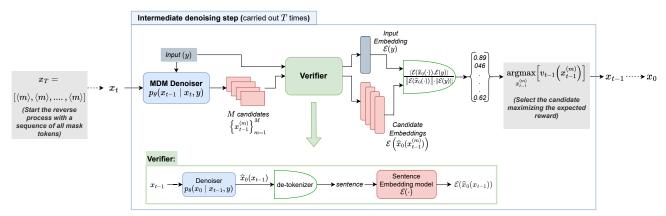


Figure 1: Diagram showing a single denoising step (i.e., the reverse process) of MDM with the soft-value diffusion decoding method paired with a pre-trained sentence-embedding-based verifier at an arbitrary timestep t. An intermediate noisy sample x_t is passed through the denoiser, and M candidates for x_{t-1} are sampled. The M candidates pass through the verifier which follows three steps: (1) x_0 — prediction for each candidate, (2) detokenize x_0 to form a sentence, and (3) pass the sentence through the embedding model to obtain sentence embedding. The input sentence embedding and the candidate embeddings are used to calculate the value function $v_{t-1}(\cdot)$ (i.e., the expected reward as defined in 3.2). The final x_{t-1} is chosen such that it maximizes the value function.

- extensive experiments with autoregressive and diffusion baselines establishing MDMs as a better alternative and highlighting the impact of inference-time scaling in MDMs (Section 5).
- empirical analysis of inference-time compute scaling methods along different scaling axes, namely denoising steps and number of samples considered during the verification step to obtain guidance signals (Section 5.2).

2 Background

2.1 Text Style Transfer

Text style transfer (TST) is a natural language generation task that involves modifying the stylistic properties of a given input text (e.g., sentiment, formality, or dialect) while preserving its underlying semantic content. TST is considered an important task in NLP since the choice of style plays a vital role in understanding user intent [14]. Formally, given a source sentence x with style s_1 , the goal of TST is to generate a new sentence x' such that $x' \approx x$ in content and exhibits a target style s_2 . The task has been explored in various supervised, unsupervised, and semi-supervised settings. When parallel corpora are available—that is, sentence pairs (x,y) where x is a sentence in style s_1 and y is its corresponding rewriting in style s_2 with the same semantic content—the task can be cast as a supervised sequence-to-sequence learning problem.

2.2 Masked Diffusion Language Models

Masked Diffusion Language Models (MDMs) define the diffusion forward and reverse processes in the discrete (or the token) space [13, 2]. The forward process iteratively replaces tokens in the sequence with mask tokens and the reverse process is learned to iteratively unmask tokens to generate new samples [30, 25].

unmask tokens to generate new samples [30, 25]. Let $x_0 = [x_0^0, x_0^1,, x_0^{L-1}]$ be a sentence where L is the length of the sentence. Let $t \in [0, 1]$ be the timestep for the diffusion process. Let V be the total vocabulary size. The forward process is defined as:

$$q_{t|0}(x_t|x_0) = \prod_{i=0}^{L-1} q_{t|0}(x_t^i|x_0^i) \quad \text{and}$$

$$q_{t|0}(x_t^i|x_0^i) := \begin{cases} \alpha_t, & x_t^i = x_0^i, \\ 1 - \alpha_t, & x_t^i = \langle m \rangle, \end{cases}$$
(1)

where x_t is the sequence at timestep t, α_t is the hyperparameter that controls the noise level (i.e., the proportion of masked tokens in the sequence) at a given timestep, $\langle m \rangle$ represents the mask token and $q_0(\cdot)$ is the data distribution. Note that the forward process is factorized across the sequence length, meaning that the noise is added independently to each token in the sequence. For timesteps s and t with $0 \le s < t \le 1$, the corresponding reverse process is defined as:

$$q_{s|t}(x_s|x_t) = \prod_{i=0}^{L-1} q_{s|t}(x_s^i|x_t) \quad \text{and}$$

$$q_{s|t}(x_s^i|x_t) = \begin{cases} \frac{t-s}{t} q_{0|t}(x_s^i|x_t), & x_t^i = \langle m \rangle, x_s^i \neq \langle m \rangle \\ \frac{s}{t}, & x_t^i = \langle m \rangle, x_s^i = \langle m \rangle \\ 1, & x_t^i \neq \langle m \rangle, x_s^i = x_t^i \end{cases} \tag{2}$$

The distribution $q_{0|t}(\cdot|\cdot)$ is learned by parameterizing it as an output of a neural network. Intuitively, a neural network is learned to predict all the masked tokens given the unmasked tokens in a noisy sentence at all possible noise levels. Formally, we write the estimated data distribution as $p_{\theta}(x_0)$ and the neural network estimate as $p_{\theta}(\cdot|x_t) \approx q_{0|t}(\cdot|x_t)$ where θ are the parameters of the neural network. To learn the network, [32, 30] derive a simplified upper-bound on the negative log-likelihood as the training objective:

$$-\log p_{\theta}(x_0) \le \int_0^1 \frac{\alpha_t'}{1 - \alpha_t} \mathbb{E} \left[\sum_{\substack{i \\ x_t^i = \langle m \rangle}} -\log p_{\theta}(x_0^i | x_t) \right] dt \quad (3)$$

The expectation is under the forward distribution $q(x_t|x_0)$. The objective is similar to the masked language modeling objective [7] since it computes the cross-entropy loss on the masked positions in

the noisy sentence. In practice, MDMs have proven their ability to scale easily [25, 26] to larger sizes, decreasing the performance gap between text diffusion models and autoregressive LMs. Additionally, unlike typical diffusion models, masked diffusion processes can be learned with a time-independent neural network, which allows us to use standard Transformer architectures with minimal changes to parameterize the reverse process.

3 Methodology

Our base method follows from section 2.2. We take an MDM pre-trained on language data and fine-tune the model on the down-stream style transfer tasks. Our fine-tuning method is tailored for sequence-to-sequence learning setups. Let x_0 be the target sequence of length L_1 (following notations from section 2.2) and y be the input conditional sequence of length L_2 . During training, we concatenate x_0 and y to form a training instance $x^{\text{inp}} = [y^1, y^2, ..., y^{L_2}, <\text{sep}>, x_0^1, x_0^2, ..., x_0^{L_1}]$, where <sep> is either a separator token or a string.

To perform the forward diffusion process, we add noise to the target part of $x^{\rm inp}$ (i.e., x_0) by following equation 1 to form $x_t^{\rm inp}$ by randomly sampling a timestep t from a uniform random distribution $\mathcal{U}(0,1)$. We keep the conditional part of $x^{\rm inp}$ (i.e., y) intact during the forward process [11, 25]. We set $\alpha_t=1-t$ following prior work [21, 32, 30]. Next, we pass $x_t^{\rm inp}$ through the neural network NN $_{\theta}$ (which means we have $p_{\theta}(x_0|x_t,y)={\rm NN}_{\theta}(x_t^{\rm inp})$). The denoiser NN $_{\theta}$ is a transformer with bi-directional self-attention. Following equation 3, we compute the loss values on the masked positions. We always compute loss only on the target sequence tokens since we only add noise to the target sequence, which makes this similar to how autoregressive LMs are instruction-tuned. More details on the training and sampling algorithms for MDMs are described in Appendix A.

3.1 Classifier-Free Guidance

Our training methodology involves using classifier-free guidance (CFG) [12], which proposes to train the neural network to estimate both conditional (i.e., $p_{\theta}(x_0|x_t,y)$) and unconditional (i.e., $p_{\theta}(x_0|x_t)$) distributions of the clean data. For learning on text data, this can be achieved by randomly setting the conditional sequence y to $\phi = [\langle m \rangle, \langle m \rangle, ..., \langle m \rangle]$ (i.e., a sequence of all mask tokens of the same length as y) during training [31]. This setting is equivalent to modeling the unconditional distribution, i.e., $p_{\theta}(x_0|x_t) = p_{\theta}(x_0|x_t, \phi)$.

After training, these distributions are used during inference to estimate the conditional log-probability of the clean data x_0 as follows:

$$\log p_{\theta,\gamma}(x_0|x_t,y) = \gamma \log p_{\theta}(x_0|x_t,y)$$

$$+$$

$$(1-\gamma) \log p_{\theta}(x_0|x_t) + c.$$

$$(4)$$

Here, γ is the classifier-free guidance scale, which offers a trade-off between sample diversity and fidelity. The constant c can be ignored since we compute probabilities using the softmax function, the results of which remain unaffected by the value of c. Typically, higher values of γ (> 1) provide more grounded outputs and result in better sample quality. At $\gamma=1$, the estimation boils down to the conditional estimate without any CFG.

Algorithm 1: The MDM reverse process with SVDD

```
Given: Denoiser \mathrm{NN}_{\theta}, sentence embedding model \mathcal{E}(\cdot), input/source sentence y, the value function v_t(\cdot) = R_{\mathcal{E},t}\left(\cdot,y\right) and total denoising steps T. Initialize: t=T while t \geq 1 do  \begin{array}{c|c} \mathrm{Compute} \ p_{\theta}(x_0|x_t,y) = \mathrm{NN}_{\theta}(x_t^{\mathrm{inp}}) \ (\approx q_{0|t}(x_0|x_t,y)) \\ \mathrm{Compute} \ p_{\theta}(x_{t-1}|x_t,y) \ \mathrm{following} \ \mathrm{eqn.} \ 2 \\ \mathrm{Sample} \ M \ \mathrm{candidates:} \ \{x_{t-1}^{(m)}\}_{m=1}^{M} \sim p_{\theta}(x_{t-1}|x_t,y) \\ x_{t-1} = \underset{x_{t-1}^{(m)}}{\mathrm{argmax}} \left[v_{t-1}(x_{t-1}^{(m)})\right], \ \mathrm{where} \ m \in \{1,2,...,M\} \\ t = t-1 \\ \mathrm{end} \\ \mathrm{return} \ x_0 \end{array}
```

3.2 Soft-Value Diffusion Decoding

The generative process of the diffusion models can be scaled beyond denoising steps by designing an evaluation mechanism for the outputs at each step. Using these external verification signals provides guidance to the generation process towards potentially better outputs. [23] denotes this as scaling along the verifier axis. In this work, we propose to design the verification step for TST tasks using off-theshelf pretrained sentence embedding models. Specifically, we score (or reward) a sampled candidate from the denoiser at each denoising step using the cosine similarity between the candidate embeddings and the input sentence embeddings. The idea is to reward the capturing of the semantic content of the input sentence in the style-transferred candidate generated from the model. Formally, the reward is defined as

$$R_{\mathcal{E},t}\left(\widehat{x}_0(x_t), y\right) := \frac{\langle \mathcal{E}(\widehat{x}_0(x_t)), \mathcal{E}(y) \rangle}{\|\mathcal{E}(\widehat{x}_0(x_t))\| \cdot \|\mathcal{E}(y)\|},\tag{5}$$

where $\mathcal{E}(\cdot)$ is an off-the-shelf pre-trained sentence embedding model and $\widehat{x}_0(x_t) \sim p_{\theta}(x_0|x_t,y)$. Along with adding an external verifier for evaluation, a search algorithm needs to be in place to find the best candidate at every denoising step based on the output from the verifier. Using the scores defined above as a reward function, we implement the Soft-Value Diffusion Decoding (SVDD) algorithm [16]. Figure 1 explains how the algorithm works with a single intermediate step of the reverse diffusion process. The algorithm derives from sampling while maximizing the reward, which boils down to sampling multiple possible candidates during a denoising step and choosing the best candidate that results in the maximum expected reward at that step. With a large enough pool of candidates, this results in an explicit reward maximization at the end of the denoising trajectory. Let $v_t(\cdot)$ be the value function denoting the expected reward at t = 0 from an arbitrary noisy state at step t'. Following the **posterior-mean-approximation (PMA)** method from [16], the value function induced by $x_{t'}$ can be approximated as $v_{t'}(x_{t'}) \approx R_{\mathcal{E},t'}(\widehat{x}_0(x_{t'}),y)$. To estimate the optimal denoising route, at every timestep, we draw M > 1 independent $x_{t'}$ samples and select the $x_{t'}$ that maximizes the value function. Algorithm 3 details the SVDD method explained above. Refer to Appendix B for more details on how SVDD with PMA leads to reward maximization.

The above formulation provides two advantages:

 As long as the denoiser is available, there is no need to explicitly train a separate reward model (i.e., the verifier, which is a pretrained sentence embedding model in our case). This is because

WikiLarge complicated to simple text style transfer									
Model	#params	#steps	BLEU ↑		SARI ↑		LENS ↑		
MDM w/o CFG		8 16 64	w/o SVDD 54.609±0.974 62.168±0.608 41.467±0.629	with SVDD (ours) 60.923±0.617 67.055±0.682 44.598+0.763	w/o SVDD 33.653±0.252 35.870±0.157 31.591±0.187	with SVDD (ours) $35.500_{\pm 0.249}$ $37.385_{\pm 0.202}$ $32.437_{\pm 0.219}$	w/o SVDD 18.794±0.804 28.552±0.765 28.983±0.552	with SVDD (ours) 22.118±0.733 31.354±0.644 30.658±0.687	
MDM with CFG	162M	8 16 64	$\begin{array}{c} 69.545 \pm 0.431 \\ 78.854 \pm 0.633 \\ 86.567 \pm 0.419 \end{array}$	$71.251_{\pm 0.327} \\ 80.208_{\pm 0.469} \\ 87.403_{\pm 0.374}$	$44.089 \pm 0.205 47.270 \pm 0.215 48.578 \pm 0.136$	$44.864_{\pm 0.139} \\ 47.949_{\pm 0.160} \\ 49.159_{\pm 0.179}$	31.623 ± 0.528 40.133 ± 0.505 46.812 ± 0.612	$33.258_{\pm 0.436}$ $41.650_{\pm 0.381}$ $47.999_{\pm 0.534}$	
GENIE	144M	64	$71.842_{\pm 0.774}$		43.268 ± 0.421		$53.863 {\scriptstyle \pm 0.868}$		
	136M	-	43.20	02±1.465	34.186 ± 0.608		43.126±1.078		
SmolLM2	360M	-	50.1	48±3.136	$38.748_{\pm 2.161}$		48.3	$48.348_{\pm 1.874}$	
Qwen2.5	490M	-	$52.268_{\pm 1.573}$		40.463 ± 0.722		$49.865_{\pm 1.053}$		

Table 1: Main results for the Wikilarge simplicity style transfer dataset. As mentioned in section 4.4, we report the mean and the standard deviation (in the subscript) across 20 independent sampling runs. ↑ indicates "higher the better" for the metric. The non-coloured cells report baseline scores, while the coloured cells represent different inference-time scaling settings.

- we have an estimate of the clean data x_0 at every denoising step, which can be directly passed to the reward model for evaluation of the expected reward.
- 2. Since the method is derivative-free, it gives the flexibility of choosing any pre-trained sentence embedding model. This is not possible with derivative-based guidance methods, where the choice of the sentence embedding model would be restricted by the requirement of having the same tokenizer as the denoiser.

4 Experimental Details

4.1 Datasets

We train and evaluate the models on two TST datasets: (1) Wiki-Large simplicity style transfer dataset² [39] and (2) Bible prose style transfer dataset³ [5]. The simplicity style transfer dataset consists of sentences and their simplified version. The corresponding training/validation/test split in the dataset is 296K/1K/0.35K⁴, respectively. The training data of the Bible prose style transfer consists of 1.5M sentence pairs representing conversions between different versions of biblical sentences as well as bible sentences in basic English style. The corresponding validation set consists of around 91K sentence pairs. For evaluation, we consider two standard test sets: conversion of public version biblical sentences (PUB-BBE) and conversion of public version biblical sentences to the American Standard Version (PUB-ASV). Each of the test sets consists of around 12K sentence pairs.

4.2 Baseline Methods

We compare our method with the following baselines:

1. Autoregressive Baselines:

- SmolLM2 [1]: a family of lightweight autoregressive LMs pretrained on 2T tokens. We perform seq-to-seq fine-tuning on the 135M and 360M parameter models from the family to create the baseline.
- **Qwen 2.5 [36]:** a family of autoregressive LMs pre-trained on 18T tokens. We perform seq-to-seq fine-tuning on the 490M parameter model from the family to create the baseline.
- ² https://github.com/XingxingZhang/dress?tab=readme-ov-file
- $^3\ https://github.com/keithecarlson/StyleTransferBibleData$
- ⁴ The test set consists of 8 references per input which can be accessed from https://github.com/cocoxu/simplification/tree/master/data/turkcorpus

- GENIE [19]: a continuous-space diffusion model that is pretrained on 160Gb of news, books, stories, and web text. Unlike MDMs, GENIE performs the diffusion processes in the embeddings of the tokens instead of directly on the discrete token sequence.
- MDM without inference-time scaling: generate outputs from the fine-tuned MDM without inference-time scaling, i.e., without using classifier-free guidance or SVDD or both.

In addition to these baselines, we compare multiple different settings during inference with the fine-tuned MDM, the details of which are discussed in section 5.1.

4.3 Training and Inference Details

For the experiments with MDM, we use pre-trained MDMs released by the work of [25]. These MDMs are pre-trained on 627B tokens from the SlimPajama dataset [33]. Specifically, we pick the MDM with 113M non-embedding parameters (167M total parameters) for fine-tuning. For the SVDD experiments, we use the MPNet-base-v25 sentence embedding model as a verifier for obtaining embedding cosine similarity rewards. For optimization, we use the AdamW optimizer [20] with $\beta_1 = 0.9$ and $\beta_2 = 0.95$. We set a small value $\epsilon=10^{-5},$ indicating the minimum possible noise level $^6.$ To schedule the learning rate, we first implement a linear warmup schedule for a small number of warmup steps, after which we perform an inverse square root decay. All training experiments are done with bfloat16 precision except for the GENIE baseline, where fp32 precision is used. While training MDMs, we use an exponential moving average of weights during the weight update with a decay rate of 0.999. In the main results (table 1 and 2), the CFG results are reported with $\gamma=1.4$ and the SVDD results are reported with M=4 candidates per denoising step. The decoding with autoregressive LM baselines is performed with nucleus sampling with p = 0.95. All of the experiments are performed on a single NVIDIA H100 GPU. Further details on hyperparameter settings for each experiment are provided in Appendix C.

4.4 Evaluation Metrics

We employ the following metrics for the evaluation of the generated outputs:

⁵ https://huggingface.co/sentence-transformers/all-mpnet-base-v2

⁶ This is done to prevent numerical overflow in loss function calculation during training as for $t \to 0$, we will have Loss $\to \infty$ given that we set $\alpha_t = 1 - t$ according to section 3.

				Public Versi	ions of Bible to Bi	ble in Basic English (PUB-BBE)			
Model	#params	#steps	s BLEU↑		ROUGE-L ↑		METEOR ↑		BERTScore ↑	
			w/o SVDD	with SVDD (ours)	w/o SVDD	with SVDD (ours)	w/o SVDD	with SVDD (ours)	w/o SVDD	with SVDD (ours)
MDM w/o CFG		8	$16.030_{\pm 0.046}$	$17.440_{\pm 0.040}$	$44.113_{\pm 0.051}$	45.642 ± 0.039	39.073 ± 0.069	$40.937_{\pm 0.031}$	87.502±0.025	87.980±0.015
MDM W/0 CFG		16 64	$22.687_{\pm 0.036}$ $26.343_{\pm 0.047}$	$24.435_{\pm 0.032}$ $27.494_{\pm 0.052}$	$50.630_{\pm 0.038}$ $52.937_{\pm 0.042}$	$52.276_{\pm 0.063}$ $54.176_{\pm 0.053}$	$46.574_{\pm 0.049}$ $48.955_{\pm 0.055}$	48.699 ± 0.054 50.439 ± 0.063	$89.715_{\pm 0.025}$ $90.715_{\pm 0.030}$	$90.205_{\pm 0.035}$ $91.118_{\pm 0.021}$
	162M		18.637 _{±0.030}	19.923 _{+0.056}	45.489 _{+0.034}	$46.901_{\pm 0.056}$	41.839 _{±0.051}	43.648±0.077	88.331 _{±0.020}	88.751 _{±0.024}
MDM with CFG		16	25.870 ± 0.030	27.058 ± 0.056	$52.340_{\pm 0.043}$	53.480 ± 0.056	50.555 ± 0.051	52.059 ± 0.044	$90.476_{\pm 0.012}$	$90.804_{\pm 0.026}$
		64	$32.704_{\pm 0.031}$	$33.304_{\pm0.029}$	$58.187_{\pm 0.038}$	$58.738_{\pm0.023}$	$57.572_{\pm 0.058}$	$58.305_{\pm0.040}$	$92.371_{\pm 0.024}$	$92.532 \scriptstyle{\pm 0.017}$
GENIE	144M	64	19.275 ± 0.064		55.505 ± 0.066		55.169 ± 0.066		$90.523_{\pm 0.009}$	
	135M	-	- 19.242 _{±0.046} - 20.391 _{±0.027}		$42.686_{\pm0.071}$		$43.664_{\pm0.045}$		90.448 _{±0.008}	
SmolLM2	360M	-			$44.291_{\pm 0.095}$		$45.339_{\pm 0.062}$		90.968 _{±0.010}	
Qwen2.5	490M	-	21.318 ± 0.045		$45.283_{\pm0.081}$		$46.524_{\pm0.169}$		91.134 _{±0.014}	
				Public Versions of	Bible to American	Standard Version of	Bible (PUB-ASV)		
		8	$28.770_{\pm 0.048}$	$30.831_{\pm 0.065}$	$56.341_{\pm 0.061}$	$58.216_{\pm 0.047}$	$51.586_{\pm0.082}$	$53.902_{\pm 0.038}$	$89.543_{\pm 0.022}$	$90.069_{\pm 0.040}$
MDM w/o CFG		16	$39.887_{\pm 0.072}$	41.725 ± 0.035	64.846 ± 0.037	$66.241_{\pm 0.035}$	$62.210_{\pm 0.051}$	63.959 ± 0.049	$92.047_{\pm 0.029}$	92.455 ± 0.029
	162M	64	$47.610_{\pm 0.038}$	48.468 ± 0.035	$70.085_{\pm0.034}$	70.788 ± 0.023	$68.469_{\pm0.033}$	$69.364_{\pm 0.022}$	$93.627_{\pm 0.023}$	$93.879_{\pm 0.036}$
	102IVI	8	33.535 ± 0.074	35.056 ± 0.041	$59.681_{\pm 0.053}$	$61.051_{\pm 0.060}$	$56.314_{\pm 0.060}$	$58.002_{\pm 0.077}$	$90.861_{\pm 0.027}$	$91.246_{\pm 0.021}$
MDM with CFG		16	$43.060_{\pm 0.055}$	$44.083_{\pm 0.042}$	66.898 ± 0.042	67.709 ± 0.053	65.378 ± 0.054	66.398 ± 0.054	$92.907_{\pm 0.022}$	$93.124_{\pm 0.023}$
		64	50.695 ± 0.034	$51.118_{\pm 0.027}$	$72.300_{\pm 0.021}$	$72.669 {\scriptstyle \pm 0.023}$	71.906 ± 0.028	$72.396_{\pm 0.028}$	$94.516_{\pm 0.020}$	$94.608_{\pm0.015}$
GENIE	144M	64	35.0	$99_{\pm 0.060}$	$71.675_{\pm 0.033}$		$71.033_{\pm 0.050}$		$92.821_{\pm 0.008}$	
	135M	-	$33.620_{\pm 0.077}$		$57.254_{\pm0.108}$		$57.138_{\pm0.156}$		$92.310_{\pm 0.014}$	
SmolLM2	360M	-	35.4	68 ± 0.072	$58.052_{\pm 0.072}$		$58.046_{\pm0.097}$		$92.497_{\pm 0.015}$	
Qwen2.5	490M	-	37.9	54 _{±0.154}	$60.706_{\pm0.037}$		$60.869_{\pm0.022}$		$92.965_{\pm 0.005}$	

Table 2: Main results for the Bible prose style transfer dataset. As mentioned in section 4.4, we report the mean and the standard deviation (in the subscript) across 8 independent sampling runs. ↑ indicates "higher the better" for the metric. The non-coloured cells report baseline scores, while the coloured cells represent different inference-time scaling settings.

- BLEU [29]: A precision-based metric that measures the overlap between machine-generated text and one or more reference texts using n-grams.
- The Following metrics are used specifically for the evaluation of the WikiLarge simplicity style transfer dataset:
 - SARI [35]: Evaluates the quality of sentence simplification by comparing system output against the source and reference texts, rewarding n-gram additions and deletions, and keeping operations that align with references.
 - LENS [24]: It is a learned reference-based metric that scores simplifications based on fluency, adequacy, and simplicity.
- The following metrics are used specifically for the evaluation of the Bible prose style transfer dataset:
 - ROUGE [18]: A recall-oriented set of metrics that evaluates the overlap of n-grams, word sequences, and word pairs between the candidate and reference texts.
 - METEOR [3]: Captures unigram matching between the model-generated text and the reference text, where unigrams can be matched based on their surface forms, stemmed forms, and meanings.
 - BERTScore [38]: Uses contextual embeddings from pretrained BERT models to compute a similarity score between candidate and reference sentences, capturing semantic similarity better than n-gram-based metrics.

During the evaluation phase, we perform multiple independent sampling runs on the test sets and report the mean and standard deviation of the metrics across the runs. For the WikiLarge dataset, we perform 20 runs, and for the Bible dataset, we perform 8 runs (due to a significantly larger test set than the former).

5 Results and Discussion

Table 1 and Table 2 list out the main results for the WikiLarge dataset and the Bible dataset, respectively. As a general trend, we

observe that the autoregressive baselines are outperformed by our diffusion baseline (i.e., GENIE). The diffusion baseline also outperforms the vanilla MDM setting (i.e., MDM without inference-time scaling). However, GENIE is pretrained with a case-insensitive tokenizer, which gives it a slight edge over other methods due to the lessened complexity of generation. This is specifically reflected in the scores for the Bible dataset, where we observe that the BLEU scores are significantly low for GENIE (as BLEU score computation involves direct n-gram matching) ⁷. Such a tokenizer design for generation can also be considered a downside since real-world scenarios would require ensuring proper capitalization during generation.

On the Bible test sets, the fine-tuned MDMs without any inference-time scaling beat the autoregressive baselines with as few as 16 steps. With the inclusion of classifier-free guidance and our proposed method with SVDD, we significantly outperform the diffusion model baseline as well. The first example of Figure 2 shows generated samples from the methods experimented with in this work. The autoregressive baselines tend to generate longer outputs and look less grounded in the input sentence. Oftentimes, we also observe out-of-context and hallucinatory behaviour from autoregressive LM generation (as highlighted in Figure 2). Such behaviour is usually not seen in MDMs due to their non-autoregressive generation method. In contrast, MDM without inference-time scaling tends to generate short outputs, but in general produces more valid tokens⁸.

On the Wikilarge test set, the fine-tuned MDMs without any inference-time scaling do not report promising simplicity scores. Qualitative analysis showed two reasons: (1) MDM produces very short outputs, possibly mistaking sentence shortening for simplicity, making the simplification inadequate in content, and (2) MDM tends to copy the input sentence almost exactly, resulting in no simplicity in the output. Both reasons lead to poor performance on simplicity metrics (specifically on LENS scores). However, such inconsistencies are vastly mitigated with inference-time scaling.

⁷ Due to direct n-gram matching with the reference, if any capitalized letter occurs in a word, it won't match with the output of GENIE. For e.g., the word "Where" will not match with "where".

⁸ Additional results on the Bible dataset with static word vector embeddings can be found in Appendix D

Input: And David said to Saul, Let no man's heart fail because of him: thy servant will go and fight with this Philistine.

Reference: And David said to Saul, Let no man's heart become feeble because of him; I, your servant, will go out and have a fight with this Philistine.

- MDM (M=4, $\gamma=1.4$, steps = 64): And David said to Saul, Let no man's heart be feeble against him: your servant will go and make war on this Philistine.
- MDM (M=1, $\gamma=1.0$, steps = 64): And David said to Saul, Let no man's heart be with him: your servant will go to war against this Philistine.
- GENIE: and david said to saul, let no man's heart be going because of him; your servant will go and make war with the whole philistine.
- SmolLM2-135M: And David said to Saul, All men go to sleep because of him; let no man go out of his mind because of him: your servant will go in flight when the Philistines are made strong.
- SmolLM2-360M: And David said, Let no man be moved in heart by my words, because your servant will go and put him to the sword.
- Qwen2.5: And David said to Saul, Let no one be overcome because of him. Your servant will go and fight with this Philistine.

Input: And Gideon taketh ten men of his servants, and doth as Jehovah hath spoken unto him, and it cometh to pass, because he hath been afraid of the house of his father, and the men of the city, to do by day, that he doth by night.

Reference: Then Gideon took ten of his servants and did as the Lord had said to him; but fearing to do it by day, because of his father's people and the men of the town, he did it by night.

- MDM (M=1, $\gamma=1.0$, steps = 16): And Gideon took ten men of his servants and did, as the Lord had said to him, and because he was the fear of made his family his and and as whom the men of the year, so he did so by night
- ullet MDM (M=4, $\gamma=1.0$, steps = 16): And Gideon took ten men of his servants, did them as the said said: and he was in fear and fear of his family and because of of men of the town span the day he he night
- MDM (M=4, $\gamma=1.4$, steps = 16): So Gideon took ten men of his servants and did them as the Lord said said: and he, because he was in fear fear come of his father's family the who of the men of the town, he did so by night night and night.
- MDM $(M=1, \gamma=1.0, steps=64)$: So Gideon took ten men of his servants and did as the Lord had said: and because he was fearing his father and his family and the men of David, days he did it by night.
- ullet MDM (M=4, $\gamma=1.0$, steps = 64): So Gideon took ten men of his servants and did as the Lord had said: and because he was fearing them to his father and his men, he did them by night
- MDM (M=4, $\gamma=1.4$, steps = 64): So Gideon took ten men of his servants and did as the Lord had said to him: and because he was full of fear of his father's family and the men of the town to do it by day, he did it by night.

Figure 2: Examples of generated outputs from PUB-BBE task. The top example displays the outputs generated from the baseline methods and the MDM (with and without inference-time scaling). Out-of-context (and sometimes hallucinatory) behaviour in autoregressive LM generations is marked with red. The bottom example displays the outputs generated by the MDM with different inference-time hyperparameters. Lack of coherence in generated sentences is marked in blue. More generated examples can be found in Figure 5 of the supplementary material.

5.1 Ablation Studies

Classifier-free guidance (CFG): Using classifier-free guidance during MDM decoding significantly improves the generation quality in both the TST tasks across different metrics (as shown in table 1 and 2). CFG improves the conditional grounding of the generated outputs using the guidance scale parameter. Intuitively, for $\gamma > 1$, we have a part of unconditional logits subtracted from the conditional logits (as per equation 5), which tends to improve the fidelity of sampled outputs from the resulting logits. Figure 3b shows the effect of increasing the classifier-free guidance scale parameter. As the value increases, the performance also increases (due to an increasing influence of the conditional logits) up to a certain point, after which it either stagnates or slightly degrades. The degradation primarily happens due to over-reliance on the conditional signal, shifting the focus from the overall fluency of the generated sentence.

Soft-value diffusion decoding (SVDD): Using SVDD with sentence embedding similarity as rewards consistently improves the generation performance of the MDM across different settings and metrics. The improvement is more significant with smaller denoising steps. The improvement can also be seen when using SVDD on top of CFG-based decoding. It is important to note that SVDD relies heavily on the inherent quality of the MDM itself. Intuitively, it tries to find the best solution amongst the pool of solutions that the MDM can generate. SVDD with our proposed reward aims to achieve this by performing semantic content maximization between

the input and the generated sentences along the denoising trajectory at every timestep. Increasing the number of possible candidates for verification leads to better final estimates.

5.2 Test-time compute scaling

The bottom example of Figure 2 shows the effect on the generation with different MDM inference settings. MDM generation without inference-time scaling often struggles with sentence coherence and tends to produce unnecessary repeated tokens. Below, we discuss in more detail the effect of scaling the finetuned MDMs along different scaling axes, where diffusion models offer the flexibility to do so.

Effect of scaling denoising timesteps: Figure 3a shows the plots for metrics vs. denoising steps. In general, increasing denoising timesteps (keeping other hyperparameters the same) leads to improving performance up to a certain point, after which it stagnates [23]. However, vanilla MDM decoding observes a slight drop in performance after moving above a certain number of denoising timesteps. This observation is slightly more pronounced in the simplicity style transfer task, where the MDM starts generating shorter outputs with more denoising steps. While shorter sentences are simpler, in a conditional setting, they often fail to capture the content, resulting in a drop in performance. The addition of classifier-free guidance allows MDMs to scale better with increasing time steps, showing trends similar to diffusion timestep scaling for images.

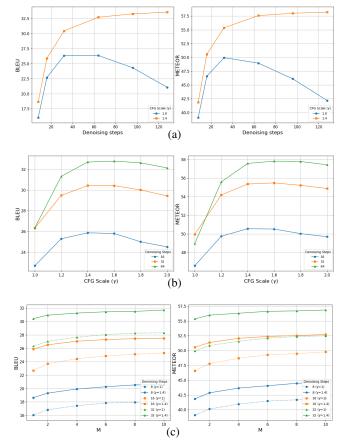


Figure 3: Plots for performance metrics vs. inference-time hyperparameters as evaluated on the PUB-BBE test set. γ is the CFG scale. (a) Shows the effect of scaling across the timestep axis with CFG (i.e., $\gamma=1.4$) and without CFG (i.e., $\gamma=1$), (b) shows the effect of the CFG scale (γ) during inference, and (c) shows the effect of scaling across the verifier axis by increasing the number of candidates (i.e., M) to be generated each step for verification. Refer to Appendix E for plots on the other two test sets.

Effect of number of candidates in an SVDD step: Figure 3c shows the effect of increasing the number of candidates verified per denoising step on the performance metrics. Similar to scaling denoising steps, increasing the number of candidates also stagnates after a certain number. Having a fixed number of candidates to be considered for verification is like a Monte-Carlo approximation for choosing the best among all possible candidates at a particular step. This means a larger number of candidates will lead to a better approximation. However, the number of forward passes to be made per step linearly increases with the number of candidates, making it essential to consider this compute-time vs. quality tradeoff carefully. The stagnation in performance after a certain number of candidates indicates that the current pool of candidates is nearly sufficient to represent the pool of all possible candidates. We also observe that SVDD improves more upon MDM inference without CFG compared to inference with CFG. Both stagnation with increasing candidates and the effect of CFG can be attributed to the diversity that the MDM sampling can exhibit. The number of verification candidates can potentially better scale the performance if the model exhibits high generation diversity, as there would be more possible solutions. Since CFG improves generation quality by sacrificing diversity, it also explains why SVDD would be slightly less effective during inference with CFG, as mentioned earlier.

6 Related Work

Text style transfer (TST) has been a widely explored NLP task consisting of many dimensions like simplicity, formality, prose, sentiment, etc. Over the years, much work has gone into these tasks, considering both supervised and unsupervised settings. Apart from the traditional seq-to-seq setups, several multi-task learning approaches have been proposed for supervised settings [28]. Also, synthetic data generation is usually performed where parallel data is scarce. For unsupervised settings, distentangled representation learning for the style and content of the text is the most popular approach [17]. The survey of [14] gives a comprehensive list of methods in both categories.

Discrete diffusion models are discrete variants of the popular diffusion modeling framework used for generating images. The framework aimed to apply diffusion modeling to discrete data modalities like text, graphs, protein sequences, etc. Due to their simplicity, two common variants widely studied for discrete diffusion are the uniform and the absorbing (or masked) variants [13, 2, 21]. Amongst these two, masked discrete diffusion has proven to be a more scalable approach and has been competitive with autoregressive LMs [30, 25, 21]. Scaling diffusion models during inference with search algorithms and external verifiers [23] has allowed diffusion-based generation to scale beyond just increasing denoising steps (which is known to quickly stagnate). These techniques include a vast literature of diffusion guidance algorithms [8, 12, 37] that enable controllability during generation. Recently, guidance mechanisms have also been extended to discrete diffusion settings [31, 27], opening potential research directions in scaling discrete diffusion language models during inference.

The work of [22] explores supervised TST using seq-to-seq continuous diffusion models [11]. While this is the only work combining diffusion models and TST (to the best of our knowledge), the work limits itself by only showcasing the application of diffusion models for TST, leaving many capabilities of diffusion models unexplored.

7 Summary, Conclusion and Future Work

In this paper, we present masked diffusion language models (MDMs) for the task of text style transfer. We explore verifier-based inferencetime scaling in MDMs and propose pre-trained sentence embedding models as verifiers. This significantly improves generated outputs by combining the proposed verifier with soft-value diffusion decoding. Additionally, we analyze the scaling behaviour of MDM inference along different scaling axes. Our method is simple and does not require additional training on top of fine-tuning the MDMs since we can directly incorporate pre-trained embedding models as verifiers. Our work not only establishes MDMs as a better alternative for TST but also emphasizes the benefits of inference-time scaling in diffusion language models. Our work can be expanded into two broad directions: (1) train external verifiers for task-specific use cases, potentially resulting in better outputs at the cost of extra training, and (2) expand the methodology for different downstream tasks by designing task-specific reward functions. We leave further exploration in these two directions for future work.

References

- [1] L. B. Allal and et al. Smollm2: When smol goes big data-centric training of a small language model. *CoRR*, abs/2502.02737, 2025.
- [2] J. Austin, D. D. Johnson, J. Ho, D. Tarlow, and R. van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, Red Hook, NY, USA, 2021. Curran Associates Inc. ISBN 9781713845393.
- [3] S. Banerjee and A. Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization, pages 65– 72, Ann Arbor, Michigan, June 2005.
- [4] T. B. Brown and et al. Language models are few-shot learners. In Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.
- [5] K. Carlson, A. B. Riddell, and D. N. Rockmore. Evaluating prose style transfer with the bible. *Royal Society Open Science*, 5, 2017. URL https://api.semanticscholar.org/CorpusID:21186239.
- [6] H. Chang, H. Zhang, L. Jiang, C. Liu, and W. T. Freeman. Maskgit: Masked generative image transformer. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 11305– 11315, 2022.
- [7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. 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, Minneapolis, Minnesota, June 2019.
- [8] P. Dhariwal and A. Nichol. Diffusion models beat gans on image synthesis. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, Red Hook, NY, USA, 2021. Curran Associates Inc. ISBN 9781713845393.
- [9] Z. Dou and Y. Song. Diffusion posterior sampling for linear inverse problem solving: A filtering perspective. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.
- [10] A. Dubey and et al. The llama 3 herd of models. *CoRR*, abs/2407.21783, 2024
- [11] S. Gong, M. Li, J. Feng, Z. Wu, and L. Kong. Diffuseq: Sequence to sequence text generation with diffusion models. In *The Eleventh Inter*national Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023. OpenReview.net, 2023.
- [12] J. Ho and T. Salimans. Classifier-free diffusion guidance. In NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications, 2021. URL https://openreview.net/forum?id=qw8AKxfYbI.
- [13] E. Hoogeboom, D. Nielsen, P. Jaini, P. Forr'e, and M. Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. In *Neural Information Processing Systems*, 2021.
- [14] D. Jin, Z. Jin, Z. Hu, O. Vechtomova, and R. Mihalcea. Deep learning for text style transfer: A survey. *Computational Linguistics*, 48(1):155– 205, Mar. 2022.
- [15] C. K. Lee, P. Jeha, J. Frellsen, P. Lio, M. S. Albergo, and F. Vargas. Debiasing guidance for discrete diffusion with sequential monte carlo. *CoRR*, abs/2502.06079, 2025.
- [16] X. Li, Y. Zhao, C. Wang, G. Scalia, G. Eraslan, S. Nair, T. Biancalani, A. Regev, S. Levine, and M. Uehara. Derivative-free guidance in continuous and discrete diffusion models with soft value-based decoding. *CoRR*, abs/2408.08252, 2024.
- [17] Y. Li, C. Li, Y. Zhang, X. Li, G. Zheng, L. Carin, and J. Gao. Complementary auxiliary classifiers for label-conditional text generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8303–8310, 2020.
- [18] C.-Y. Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.
- [19] Z. Lin, Y. Gong, Y. Shen, T. Wu, Z. Fan, C. Lin, N. Duan, and W. Chen. Text generation with diffusion language models: a pre-training approach with continuous paragraph denoise. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023
- [20] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019. OpenReview.net, 2019.
- [21] A. Lou, C. Meng, and S. Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In Forty-first International

- Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024. OpenReview.net, 2024.
- [22] Y. Lyu, T. Luo, J. Shi, T. Hollon, and H. Lee. Fine-grained text style transfer with diffusion-based language models. In *Proceedings of the* 8th Workshop on Representation Learning for NLP (RepL4NLP 2023), pages 65–74, Toronto, Canada, July 2023.
- [23] N. Ma, S. Tong, H. Jia, H. Hu, Y. Su, M. Zhang, X. Yang, Y. Li, T. S. Jaakkola, X. Jia, and S. Xie. Inference-time scaling for diffusion models beyond scaling denoising steps. *CoRR*, abs/2501.09732, 2025. doi: 10. 48550/ARXIV.2501.09732. URL https://doi.org/10.48550/arXiv.2501.09732.
- [24] M. Maddela, Y. Dou, D. Heineman, and W. Xu. LENS: A learnable evaluation metric for text simplification. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 16383–16408, Toronto, Canada, July 2023.
- [25] S. Nie, F. Zhu, C. Du, T. Pang, Q. Liu, G. Zeng, M. Lin, and C. Li. Scaling up masked diffusion models on text. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=WNvvwK0tut.
- [26] S. Nie, F. Zhu, Z. You, X. Zhang, J. Ou, J. Hu, J. Zhou, Y. Lin, J. Wen, and C. Li. Large language diffusion models. *CoRR*, abs/2502.09992, 2025. doi: 10.48550/ARXIV.2502.09992. URL https://doi.org/10.48550/arXiv.2502.09992.
- [27] H. Nisonoff, J. Xiong, S. Allenspach, and J. Listgarten. Unlocking guidance for discrete state-space diffusion and flow models. *CoRR*, abs/2406.01572, 2024. doi: 10.48550/ARXIV.2406.01572. URL https://doi.org/10.48550/arXiv.2406.01572.
- [28] X. Niu, S. Rao, and M. Carpuat. Multi-task neural models for translating between styles within and across languages. In E. M. Bender, L. Derczynski, and P. Isabelle, editors, *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1008–1021, Aug. 2018.
- [29] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In P. Isabelle, E. Charniak, and D. Lin, editors, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002.
- [30] S. S. Sahoo, M. Arriola, Y. Schiff, A. Gokaslan, E. Marroquin, J. T. Chiu, A. Rush, and V. Kuleshov. Simple and effective masked diffusion language models. In *Annual Conference on Neural Information Processing Systems* 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 15, 2024, 2024.
- [31] Y. Schiff, S. S. Sahoo, H. Phung, G. Wang, S. Boshar, H. Dalla-torre, B. P. de Almeida, A. M. Rush, T. PIERROT, and V. Kuleshov. Simple guidance mechanisms for discrete diffusion models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=i5MrJ6g5G1.
- [32] J. Shi, K. Han, Z. Wang, A. Doucet, and M. Titsias. Simplified and generalized masked diffusion for discrete data. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.
- [33] D. Soboleva, F. Al-Khateeb, R. Myers, J. R. Steeves, J. Hestness, and N. Dey. SlimPajama: A 627B token cleaned and deduplicated version of RedPajama, 2023. URL https://huggingface.co/datasets/cerebras/ SlimPajama-627B.
- [34] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.
- [35] W. Xu, C. Napoles, E. Pavlick, Q. Chen, and C. Callison-Burch. Optimizing statistical machine translation for text simplification. *Trans*actions of the Association for Computational Linguistics, 4:401–415, 2016.
- [36] A. Yang and et al. Qwen2.5 technical report. CoRR, abs/2412.15115, 2024. doi: 10.48550/ARXIV.2412.15115. URL https://doi.org/10. 48550/arXiv.2412.15115.
- [37] H. Ye and et al. TFG: Unified training-free guidance for diffusion models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.
- [38] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi. Bertscore: Evaluating text generation with BERT. In 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020. OpenReview.net, 2020.
- [39] X. Zhang and M. Lapata. Sentence simplification with deep reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 584–594.

Appendices

Training and Sampling Algorithms

Algorithm 2 gives the training algorithm for MDMs. For each training sample x_0 , we sample a random t, run the forward masked diffusion process, and compute the loss function (i.e., the cross-entropy loss on the masked positions). After loss function computation, we take the optimization step by computing the loss gradients.

Algorithm 2: MDM Training Algorithm

```
Given: Denoiser NN_{\theta}, input/source sentence y, mask token
   \langle m \rangle.
repeat
        Forward process: x_0 \sim \text{data}, t \sim \mathcal{U}(0, 1), and
           x_t \sim q_{t|0}(x_t|x_0)
       \begin{aligned} x_t &\sim q_{t|0}(x_t|x_0), \\ \textbf{Compute:} \ p_{\theta}(x_0^i|x_t,y) &= \text{NN}_{\theta}(x_t,y) \\ \textbf{Compute loss:} \ \mathcal{L}_{\text{MDM}} &= -\sum\limits_{\substack{x_t^i = \langle m \rangle}} \log p_{\theta}(x_0^i|x_t,y) \end{aligned}
        Gradient descent with \nabla_{\theta} \mathcal{L}_{\text{MDN}}
until convergence;
```

Algorithm 3 gives the sampling algorithm for MDMs. In our work, we use the greedy decoding algorithm proposed by [6]. Let k be the number of tokens to be unmasked at an arbitrary timestep. The algorithm then chooses the top - k tokens predicted with the highest probability using $p_{\theta}(x_0|x_t)$. The prediction $p_{\theta}(x_0|x_t)$ can also be done using classifier-free guidance as described in Section 3.1 of the main content.

Algorithm 3: MDM Sampling Algorithm

Given: Denoiser NN_{θ} , input/source sentence y, Total denoising steps = T, generation sequence length L, mask token $\langle m \rangle$.

Define: c_i as the probability of a token x_0^i as assigned by the distribution $p_{\theta}(x_0^i|x_t)$.

Initialize
$$x_T$$
 as a sequence of all masked tokens of length L . for $t=1,\frac{T-1}{T},\frac{T-2}{T},\dots,\frac{1}{T}$ do
$$\begin{vmatrix} s=t-\frac{1}{T} \\ k=\lfloor L(1-s) \rfloor \\ p_{\theta}(x_0^i|x_t,y)=\mathrm{NN}_{\theta}(x_t,y) \end{vmatrix}$$
 for $i=[1,2,\dots,L]$ do
$$\begin{vmatrix} \mathbf{if}\ x_t^i\neq\langle m\rangle\ \mathbf{then} \\ x_0^i=x_t^i\ \mathrm{and}\ c_i=1 \end{vmatrix}$$
 else
$$\begin{vmatrix} x_0^i=\mathrm{argmax}_j\ p_{\theta}(x_0^i|x_t,y)\ and\ c_i=p_{\theta}(x_0^i|x_t,y)_{x_0^i} \end{vmatrix}$$
 end
$$\begin{vmatrix} \mathbf{if}\ c_i\in top\text{-}k\ \left(\{c_l\}_{i=1}^{L-1}\right)\ \mathbf{then} \\ x_s^i=x_0^i \end{vmatrix}$$
 end end return x_0

The SVDD algorithm using the proposed sentence-embedding verifier can be easily incorporated into the algorithm by plugging in the candidate selection process into the part where we compute x_s from x_t . Specifically, at each denoising step, we obtain multiple samples for x_s and follow the SVDD algorithm (described in Section 3.2 and Algorithm 1 of the main content) to select the best x_s candidate for that denoising step.

В **Soft-Value Diffusion Decoding**

Soft-value diffusion decoding is based on the principle of generation via reward maximization while preserving the original distribution that was initially trained for generating a particular distribution. Let $p_{\theta}(\cdot)$ be the approximate data distribution induced by a pre-trained neural network with parameters θ . Formally, we aim to sample from a new distribution $p_{\alpha}(\cdot)$ which is defined as

$$p_{\alpha}(x) := \underset{p}{\operatorname{argmax}} \left[\mathbb{E}_{x \sim p(\cdot)} \left(r(x) \right) - \alpha \operatorname{KL} \left(p(\cdot) || p_{\theta}(\cdot) \right) \right]. \tag{6}$$

Here, the first term corresponds to reward maximization where $r(\cdot)$ is the reward function, and the second term aims to preserve the pretrained knowledge about the distribution learned by $p_{\theta}(\cdot)$. The definition can be reduced as follows:

$$p_{\alpha}(x) = \underset{p}{\operatorname{argmax}} \left[\mathbb{E}_{x \sim p(\cdot)} \left(\alpha \log e^{r(x)/\alpha} \right) - \\ \mathbb{E}_{x \sim p(\cdot)} \left(\alpha \log \frac{p(x)}{p_{\theta}(x)} \right) \right],$$

$$\propto \underset{p}{\operatorname{argmax}} \left[e^{r(x)/\alpha} p_{\theta}(x) \right]. \tag{7}$$

We define a value function from the perspective of Masked Diffusion Language Models and show how soft-value diffusion decoding approximates the optimal denoising trajectory. The value function $v_t(\cdot)$ gives the expected reward (in the future after the reverse process is finished) at t = 0 for an arbitrary noisy state t. Based on equation 7, we formally write

$$v_t(\cdot) := \alpha \log \mathbb{E}_{x_0 \sim p_{\theta}(x_0|x_t)} \left[e^{r(x_0)/\alpha} | x_t = \cdot \right]. \tag{8}$$

Here, $\mathbb{E}_{x_0 \sim p_{\theta}(x_0|x_t)}(\cdot)$ is induced by $p_{\theta}(x_t|x_{t+1})$. Assuming a time discretization of $t \in \{T + 1, T, ..., t, t - 1, ..., 1\}$ in the diffusion sampling process, we can write $p_{\alpha}(x_0) =$ $\int \prod_{t=T}^{0} p_{\alpha}(x_{t}|x_{t+1}) dx_{1:T}$. The aim is to sample from the denoising trajectory for reward maximization. To do this, [16] performs importance sampling at every denoising step as follows:

$$p_{\alpha}^{\star}(x_{t}|x_{t+1}) \approx \sum_{m=1}^{M} \frac{\omega_{t}^{(m)}}{\sum_{t=1}^{M} \omega_{t}^{(i)}} \, \delta(x_{t}^{(m)}),$$
 where $\{x_{t}^{m}\}_{m=1}^{M} \sim p_{\theta}(x_{t}|x_{t+1}) \text{ and } \omega_{t}^{(m)} := e^{v_{t}(x_{t}^{(m)})/\alpha}.$ (9)

Here, M is the number of candidate samples considered for x_t to approximate the reward-maximizing denoising trajectory, and $\delta(\cdot)$ is the Kronecker delta function. The importance sampling weights ω_t are defined in terms of the value function (i.e., the expected reward). This allows us to steer the denoising trajectory towards generation with higher rewards. In other words, sampling x_t from x_{t+1} is broken down into two steps: (1) sample $M x_t$ candidates and (2) sample an index $\xi \in [1,2,...,M]$ with $[\omega_t^{(1)},\omega_t^{(2)},...,\omega_t^{(M)}]$ as unnormalized probability mass values thereby giving us $x_t^{(\xi)}$ as the final selected candidate.

The value function can be approximated using the posterior-mean-approximation (PMA) method [16]. This method defines a reward function on x_0 (i.e., the clean data estimate) and then computes the value function $v_t(\cdot)$ by first sampling an x_0 estimate from $p_\theta(x_0|x_t)$ and then passing the estimate to the reward function.

$$v_t(x_t) := \alpha \log \mathbb{E}_{x_0 \sim p_{\theta}(x_0 \mid x_t)} \left[e^{r(x_0)/\alpha} \mid x_t \right]$$

$$\approx \alpha \log \left(e^{r(\widehat{x}_0(x_t))/\alpha} \right)$$

$$= r\left(\widehat{x}_0(x_t) \right). \tag{10}$$

Here, $\widehat{x}_0(x_t) \sim p_\theta(x_0|x_t)$ is used to approximate the value function. Since the reward function takes an estimate of clean data, it is possible to use a pre-trained model as a reward function directly without extra training.

C Training and Hyperparameter Details

While MDM and GENIE (baseline) are diffusion models, their training and sampling regimes differ significantly. MDMs define the masked diffusion process in the discrete (or token) space while GENIE defines the Gaussian diffusion process in the token embedding (i.e., continuous) space. Also, MDMs are trained in a continuous-time setting (i.e., the diffusion model is trained in the timestep range of (0, 1]), which allows us to have flexibility in the discretization of the timestep interval during generation. In contrast, GENIE defines the diffusion process in discrete time, for which it needs to fix the maximum number of discrete timesteps during training. While it is possible to sample in fewer time steps than defined in training (for instance, using the DDIM sampling algorithm [34]), GENIE does not employ such methods for text generation, which restricts its sampling process to the same number of time steps as training.

C.1 Architecture

For MDM, we follow [25] for the neural network architecture, which is a transformer encoder. Unlike the usual diffusion model architectures, the architecture does not incorporate extra timestep embedding. The model configuration is as follows: 12 layers, 768 hidden dimension size, 12 attention heads, and 3072 intermediate dimension size. The autoregressive baselines follow the transformer decoder architecture. For GENIE, we have an encoder-decoder architecture with the following configuration for both encoder and decoder: 6 layers, 768 hidden and intermediate dimensions, and 128 embedding dimensions.

D Additional Experiments with Static Word Vector Embeddings

Alongside the usage of semantic contextual embeddings from sentence embedding models, we also performed experiments with static word vector embeddings. To perform this experiment, we take fast-text embeddings of each word in the sentence and average them to use as a sentence embedding. Table 4 shows the results on the bible dataset. We observe that the overall performance is not very far behind that of the contextual sentence embeddings.

Contextual sentence embeddings are better at capturing the semantic content of a sentence than averaged static word vector embeddings. However, they also tend to embed style along with the semantics of the sentence. In contrast, static word vector embeddings are

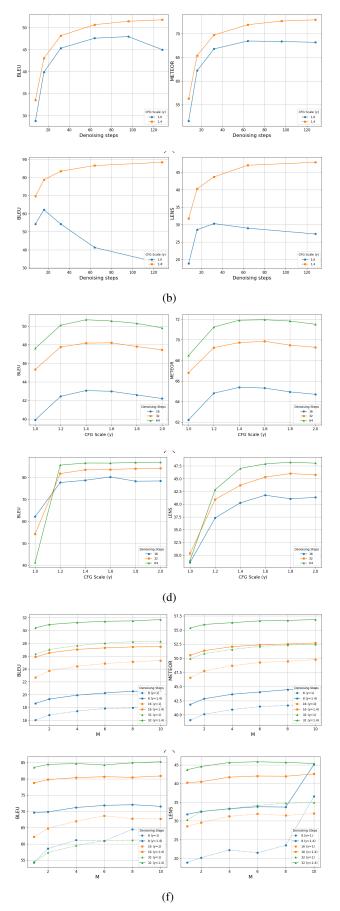


Figure 4: Additional plots on inference-time scaling and the effect of CFG scale on the model's performance. Plots (a), (c), and (e) correspond to the PUB-ASV test set, and plots (b), (d), and (f) correspond to the Wikilarge test set.

Model	Dataset	Training steps	Batch size	Training Precision	Max Grad Norm	Warmup steps	Max LR	LR scheduler	Gradient accumulation steps	weight decay
MDM	Bible	30000	256		1.0	1000	3e-4		1	0.01
	Wikilarge	34000	256		1.0	1000	3e-4		1	0.05
SmolLM2-135M	Bible	17500	256	bf16	1.0	500	2e-5	Inverse square root decay with	1	0.01
	Wikilarge	22000	128		1.0	500	4e-5	warmup	2	0.01
SmolLM2-360M	Bible	18000	128		1.0	1000	5e-6		2	0.05
	Wikilarge	14000	128		1.0	500	3e-5		2	0.1
Owen2.5-0.5B	Bible	16000	128		1.0	500	1e-6		2	0.1
	Wikilarge	32000	128		1.0	500	2e-6		2	0.1
GENIE	Bible	60000	512	fp32	-	7200	1e-4	Linear schedule	1	0.0
	Wikilarge	50000	256		-	7200	5e-5	with warmup	1	

Table 3: Hyperparameter settings for the experiments reported in the paper.

PUB-BBE						
Steps	BLEU	ROUGE-L	METEOR	BERTScore		
8	$19.796_{\pm 0.048}$	$46.741_{\pm 0.064}$	43.522 ± 0.074	$88.663_{\pm 0.025}$		
16	$27.045_{\pm 0.039}$	$53.256_{\pm 0.033}$	$51.810_{\pm 0.046}$	$90.778_{\pm 0.028}$		
64	$33.252_{\pm0.032}$	$46.901_{\pm 0.056}$	43.648 ± 0.077	$88.751_{\pm 0.024}$		
		PUB-ASV				
8	$35.075_{\pm 0.035}$	$60.969_{\pm 0.046}$	$58.010_{\pm 0.041}$	$91.196_{\pm 0.020}$		
16	$44.071_{\pm 0.049}$	$67.668_{\pm 0.037}$	$66.420_{\pm 0.050}$	$93.102_{\pm 0.019}$		
64	$51.103_{\pm 0.032}$	$72.661_{\pm 0.039}$	72.405 ± 0.035	$94.587_{\pm 0.036}$		

Table 4: Additional results on the Bible dataset using static word vector embeddings. All results are computed with a guidance scale of 1.4.

style agnostic but do not capture semantics very well. As both settings have their own flaws, it can be an interesting exploration of how both can be combined. Another way can be to fine-tune contextual sentence embeddings to be style agnostic, given parallel style transfer data, which may potentially improve the performance.

E Additional Plots on Inference-time Scaling

In Figure 4, we include the plots on inference-time scaling and the effect of CFG scale for PUB-ASV and the Wikilarge test sets (similar to PUB-BBE plots in Figure 3 of the main content). We mostly observe trends similar to those of the PUB-BBE dataset. Without CFG, the performance on the WikiLarge dataset drops much more quickly than the Bible test sets as we scale denoising steps. Interestingly, we observe a sharp spike in simplicity scores with 8 denoising steps and 10 candidate generations with the SVDD algorithm (as seen in Figure 4f).

PUB-BBE Generated outputs:

Input: And Jehovah was with Judah; and drove out the inhabitants of the hill-country; for he could not drive out the inhabitants of the valley, because they had chariots of iron.

Reference: And the Lord was with Judah; and he took the hill-country for his heritage; but he was unable to make the people of the valley go out, for they had war-carriages of iron.

- MDM (M=4, $\gamma=1.4$, steps = 64): And the Lord was with Judah, driving out the people of the hill-country; for he was not able to overcome the people of the valley, because they had war-carriages of iron.
- MDM $(M=1, \gamma=1.0, \text{ steps}=64)$: And the Lord was with Judah, driving out the people of the hill-country; for he was not able to send the people of the valley, because they had their carriages of iron
- GENIE: and the lord was with judah, driving out the people of the hill-country; for they had a desire not able to send out the people of the valley, because they were war of iron.
- SmolLM2-135M: And the Lord was with Judah, and made them the heritage of the nations, because of the fighting-men of the hill-country; for they were able to overcome the fighting men of the valley, because they had the use of iron in their armies.
- SmolLM2-360M:And the Lord kept back from Judah all those who had taken cover in the hills: for he could not get the hill-country to go away from him, because they had iron chariots.
- Qwen2.5: And the Lord was with Judah; and the men of the hill-country were driven out; because he was not able to drive out the inhabitants of the valley, because of the iron-horned men.

PUB-ASV Generated outputs:

Input: The raiders went out from the camp of the Philistines in three divisions. One division turned on the road to Ophrah toward the land of Shual.

Reference: And the spoilers came out of the camp of the Philistines in three companies: one company turned unto the way that leadeth to Ophrah, unto the land of Shual.

- MDM $(M=4,\ \gamma=1.4,\ \text{steps}=64)$: And the destroyers went out from the camp of the Philistines in three hosts: and one course turned by the way of Ophrah, unto the land of Shual.
- ullet MDM $(M=1,\;\gamma=1.0,\;{
 m steps}=64)$: And the archers went out from the camp of the Philistines in three companies: and one course turned toward the way of Ophrah, unto the land of Shual.
- GENIE: and the raiders went forth out of the camp of the philistines in three divisions; one division turned the way to ophrah toward the land of shual.
- SmolLM2-135M: And the robbers fled from the camp of the Philistines unto three companies, which turned to the way to Ophrah, toward the land of Susan.
- SmolLM2-360M: And the camp of the Philistines had passed forth as three companies; and one passed the way to Ophrah, and another went up into the wilderness by the way to Shalem.
- Qwen2.5: And the raiders went forth from the camp of the Philistines in three divisions. One division turned on the way to Ophrah, to the land of Shaual, and overthrew their cities.

Wikilarge Generated outputs:

Input: if there are no strong land use controls , buildings are built along a bypass , converting it into an ordinary town road , and the bypass may eventually become as congested as the local streets it was intended to avoid.

Reference: if there are no strong land use controls , the bypass may eventually become as congested as the local streets .

- ullet MDM $(M=4,\;\gamma=1.4,\;{
 m steps}$ = 64): if there are no strong land use controls , buildings are built along a bypass , changing it into an ordinary town road .
- \bullet MDM (M=1 , $\,\gamma=1.0$, steps = 64): the bypass .
- GENIE: if there are no strong land use controls, buildings are built along a bypass and turn it into an ordinary town road. the bypass may also go, and the it may eventually become as congested as the local streets it intended to avoid.
- SmolLM2-135M: In a city without strong land use controls , buildings are built along a bypass , changing it into an ordinary town road , and the bypass may eventually become as congested as the local streets it was intended to avoid .
- SmolLM2-360M:If there are no strong land use controls , buildings are built along a bypass , turning a road into a street .
- Qwen2.5: In addition , if there are no strong land use controls , buildings are built along a bypass , converting it into an ordinary town road .

Figure 5: Generated output examples for all the test sets considered in the evaluation. Out-of-context (and sometimes hallucinatory) behaviour in autoregressive LM generations is marked with red. For Wikilarge generations, we see vanilla MDMs without inference-time scaling outputs an extremely small and inadequate output. As discussed in the main content, this may happen due to shorter sentences being mistaken for simplicity.