Diffusion Language Models Generation Can Be Halted Early

Anonymous ACL submission

Abstract

Diffusion Language models (DLMs) are a promising avenue for text generation due to their practical properties on tractable controllable generation. They also have the advantage of not having to predict text autoregressively. However, despite these notable features, DLMs have not yet reached the performance levels of their Autoregressive counterparts. One of the ways to reduce the performance gap between these two types of language models is to speed up the generation of DLMs. Therefore, we propose a pioneering methodology to address this issue in this work. It enables the execution of more generation steps within a given time frame, potentially leading to higher-quality outputs. Specifically, our methods estimate DLMs completeness of text generation and allow adaptive halting of the generation process. We test and refine our methods on Plaid, SSD, and CDCD DLMs and create a cohesive perspective on their generation workflows. Finally, we confirm that our methods allow halting Plaid, SSD, and CDCD models and decrease the generation time by 10-40% without a drop in the quality of model samples.

1 Introduction

Language Models (LMs) are essential Natural Language Processing (NLP) tools. The two primary methods of training LMs for NLP are autoregressive training (Radford et al., 2019; Raffel et al., 2020; Chowdhery et al., 2022) and masked language modeling (Devlin et al., 2019; He et al., 2020; Liu et al., 2019; Lan et al., 2020).

The exploration of alternative models, such as Diffusion Models (Ho et al., 2020; Song et al., 2020), is a promising avenue for research as diffusion allows native non-causal conditioning and simplified controllable generation methods (Nichol et al., 2022). In recent works, with models such as "Diffusion LM" and Plaid (Li et al., 2022; Gulrajani and Hashimoto, 2023), Simplex-based Diffusion Language Model (SSD) (Han et al., 2023), GENIE (Lin et al., 2022), and Continuous Diffusion for Categorical Data (CDCD) (Dieleman et al., 2022) being introduced, we can see an emerging interest for using Diffusion Models in text generation.

A crucial distinction between Autoregressive LMs and Diffusion Language Models (DLMs) lies in their modeling approaches. Autoregressive LMs predominantly adhere to the common probabilistic model. In contrast, DLMs exhibit substantial divergence in their application for modeling categorical data. When exploring DLMs, it is essential to consider the lack of connectivity between such models. The majority of comparisons between them have primarily focused on evaluating sample quality.

While it is essential to study the sample quality of DLMs, it does not further our understanding of the differences between these models. This work addresses this issue and evaluates various DLMs with a unified view of their generation process. Given this unified view, we study the dynamics of the generation process within different DLMs and focus on the changes in the samples during that process.

The main contributions of this paper can be summarized as follows:

- We showed that the generation process of most DLMs for general text generation can be halted, which makes it possible to implement an early, faster sample generation without compromising quality.
- To the best of our knowledge, we were the first to evaluate DLMs with adaptive Early Exiting (Graves, 2016). In this paper, we introduced three adaptive criteria inspired by the ones used for text classification (Liu et al., 2020; Zhou et al., 2020; Gao et al., 2023).
- We evaluated these criteria and provided empirical evidence of their efficiency. This study

highlights the efficacy of our approach and its potential to enhance text generation by employing diffusion models. In future works, the methodology used in this paper can be developed even further in order to understand better and evaluate newly trained DLMs.

2 Related Work

2.1 Diffusion Language Models

When applied to discrete data, diffusion models have demonstrated promising results in image generation and captioning (Chen et al., 2022).

Within NLP, diffusion models have also been successfully integrated into sequence-to-sequence tasks (Savinov et al., 2021; Reid et al., 2023; Gong et al., 2023; Yuan et al., 2022; Lin et al., 2022). Despite their performance being on par with non-autoregressive models, most diffusion models employ an Encoder-Decoder architecture. For unconditional language modeling, this approach is not ideal. Although it is possible to modify probabilistic models to accommodate unconditional text generation, the supporting evidence for their efficacy in this area is sparse. It is generally acknowledged that non-autoregressive models are more adept at dealing with conditional text generation tasks, such as machine translation, compared to unconditional text modeling (Gu et al., 2018).

The "Diffusion LM" proposed by Li et al. (2022) aimed to establish a generalized LM capable of unconditional sampling. This model was evaluated based on its capability for controlled, classifier-guided text generation. However, it is worth noting that, unlike other pre-trained models, the "Diffusion LM" was not trained on large datasets. Moreover, its authors did not share any pre-trained weights, making it necessary to train the model from scratch to compare its performance with other methods. Conversely, a significant advantage of the Simplex-based Diffusion Language Model (SSD) and Plaid models (Han et al., 2023; Gulrajani and Hashimoto, 2023) is that they are available in open access and are pre-trained on extensive text datasets.

Both Self-conditioned Embedding Diffusion (SED) and Continuous Diffusion for Categorical Data (CDCD) have utilized large datasets for pre-training their Diffusion LMs (Strudel et al., 2023; Dieleman et al., 2022). However, neither of them has provided trained model weights or source code.

These diffusion models are appealing to use for comparison due to the different approaches used for training them. For instance, while CDCD utilizes a score interpolation objective, SSD works with a simplex-based method. On the other hand, Plaid is defined with a Variational Lower Bound objective (Kingma and Welling, 2014; Kingma et al., 2021).

2.2 Early Exiting Methods

The early exit technique is an approach for reducing computational load (Graves, 2016). It especially benefits transformer-based architectures, where intermediate hidden states maintain consistent shapes across layers. As a result, early exiting has become a standard technique for downstream tasks with pre-trained LMs (Zhou et al., 2020; Liu et al., 2020; Balagansky and Gavrilov, 2022; Gao et al., 2023).

3 Preliminaries

This section will briefly describe the parts essential for understanding various DLMs – CDCD, Plaid, and SSD. While each framework contains many nuances necessary to train and generate samples, we will cover details on loss evaluation and how architecture is defined to evaluate this loss. More concretely, all these models model a categorical distribution over tokens, making it possible to evaluate early exiting with them.

For each model, we start with a sequence of tokens $\boldsymbol{x} \in V^l$, where V is a vocabulary set and l is the length of a sequence.

These tokens are embedded with the embedding matrix $\boldsymbol{E} \in \mathbb{R}^{|V| \times d}$, where d is an embedding size, and produce $\boldsymbol{X}_0 \in \mathbb{R}^{l \times d}$. Subscript 0 here states that these embeddings do not contain noise, commonly used in the diffusion probabilistic model. Models then operate with the noised embeddings $\boldsymbol{X}(t)$ where the noise amount depends on timestep t.

Also, on top of these models, it is expected to see a layer producing a categorical distribution over possible tokens $p(\boldsymbol{x}|\cdots)$, conditioned on arbitrary entities (usually $\boldsymbol{X}$ and t). It is important to note that talking about a distribution over tokens is interchangeable with discussing a distribution over their corresponding embeddings since each token maps to a specific embedding vector $\boldsymbol{e}$ in $\mathbb{R}^d$ from the embedding matrix $\boldsymbol{E}$, and the process can be reversed. Throughout our discussion, this distribution will also be referred to as $p(\boldsymbol{e}|\cdots)$.

3.1 Continuous Diffusion for Categorical Data

Continuous Diffusion for Categorical Data (CDCD) operates with noised input embeddings $\boldsymbol{X}(t) \in \mathbb{R}^{l \times d}$, where the amount of noise depends on timestep $t \in [0;1]$. This process predicts a denoised sequence with categorical distribution $p(\boldsymbol{x}|\boldsymbol{X}(t),t)$. This distribution is obtained by predicting logits of shape $\mathbb{R}^{l \times |V|}$ and applying the softmax function. Subsequently, cross-entropy loss is applied to estimate $p(\boldsymbol{x}|\boldsymbol{X}(t),t)$; i.e.,

$$\mathcal{L}_{CDCD} = -\log\big(p(\boldsymbol{x}|\boldsymbol{X}(t),t)\big).$$

An estimation of the score function is calculated as $\hat{\boldsymbol{S}}(\boldsymbol{X}(t),t) = \frac{\hat{\boldsymbol{X}}_0(\boldsymbol{X}(t),t)-\boldsymbol{X}(t)}{t^2}$, where $\hat{\boldsymbol{X}}_0(\boldsymbol{X}(t),t) = \mathbb{E}_{p(\boldsymbol{x}|\boldsymbol{X}(t),t)}\big[[\boldsymbol{E},\dots,\boldsymbol{E}]\big] := \mathbb{E}_{p(\boldsymbol{e}|\boldsymbol{X}(t),t)}\big[[\boldsymbol{E},\dots,\boldsymbol{E}]\big] \in \mathbb{R}^{l \times d}$ represents the estimation of the denoised embeddings based on their probabilities (Karras et al., 2022). $\hat{\boldsymbol{S}}(\boldsymbol{X}(t),t)$ then could be passed to an arbitrary ODE solver to obtain samples from the model.

3.2 Plaid

Plaid uses simple loss derived from Variational Lower Bound objective (Kingma and Welling, 2014; Kingma et al., 2021; Gulrajani and Hashimoto, 2023)

$$\mathcal{L}_{\text{VLB}} = -\frac{1}{2}\mathbb{E}_{t,\boldsymbol{Z}_t}\Big[\frac{d}{dt}\frac{1}{\sigma^2(t)}||\boldsymbol{X}_0 - \hat{\boldsymbol{X}}_0(\boldsymbol{Z}_t)||_2^2\Big].$$

Here $\hat{\boldsymbol{X}}_0(\boldsymbol{Z}_t) \in \mathbb{R}^{l \times d}$ is an estimation of denoised embeddings from noise $\boldsymbol{Z}_t$ at time step t, $t \sim U[0;1]$, $\boldsymbol{Z}_t \sim q(\boldsymbol{Z}_t|\boldsymbol{x})$ is a distribution of forward process defined as $q(\boldsymbol{Z}_0|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{X}_0;\sigma^2(0))$, $q(\boldsymbol{Z}_t|\boldsymbol{Z}_s) = \mathcal{N}(\boldsymbol{Z}_s;\sigma^2(t) - \sigma^2(s))$. $\sigma^2(t)$ is modelled following Kingma et al. (2021).

Notably, while $\hat{\boldsymbol{X}}_0$ could be modeled in continuous space, doing so will require a model to remember initial embeddings, which is redundant. Instead, plaid uses a categorical reparametrization similar to one used with Section 3.1 and directly learns $p(\boldsymbol{e}|\boldsymbol{Z}_t)$ to estimate $\hat{\boldsymbol{X}}_0(\boldsymbol{Z}_t) = \mathbb{E}_{p(\boldsymbol{e}|\boldsymbol{Z}_t)}\big[[\boldsymbol{E},\dots,\boldsymbol{E}]\big]$.

3.3 Simplex-based Diffusion Language Model

Simplex-based Diffusion Language Model (SSD) is a third Diffusion LM tested with unconditional text generation for general language modeling.

Starting with token sequence $\boldsymbol{x}$, SSD firstly defines the operation for almost-one-hot encodings of $\boldsymbol{x}$, namely *logits generation*. For token $\boldsymbol{x}_i$, its continuous representation is defined as $\boldsymbol{X}_{i,j} = K$ if $\boldsymbol{x}_i = \boldsymbol{V}_j$, and $\boldsymbol{X}_{i,j} = -K$ otherwise. $K \in \mathbb{R}^+$ here is a hyperparameter, and $\boldsymbol{V}$ is a vocabulary.

Then, for the forward process of the diffusion with sequence $\boldsymbol{x}$, we evaluate logit generation for tokens $\boldsymbol{x}_{c:l}$, where c is a context length. Noise is progressively added to almost-one-hot encodings of the text $\boldsymbol{X}_{c:l}(t)$, leading to normal distribution across logits at the end.

The model is then trained with a loss

$$\mathcal{L}_{\text{SSD}} = \mathbb{E}_{c,t}\Big[\sum_{j=c}^{l-1} -\log\big(p(\boldsymbol{x}_j|\boldsymbol{X}_{c:l}(t),\boldsymbol{x}_{<c})\big)\Big],$$

where $c \sim U(1;l)$ is the length of context for a generation, and $t \sim U(1;T)$, and $p(\boldsymbol{x}_j|\boldsymbol{X}_{c:l}(t),\boldsymbol{x}_{<c})$ is a categorical distribution over tokens from vocabulary.

4 Early Exiting with DLMs

While CDCD, Plaid, and SSD define different views on training DLMs, they share a similarity in how distribution on denoised text is defined. More concretely, they all define a categorical distribution over possible embeddings (and thus over possible tokens).

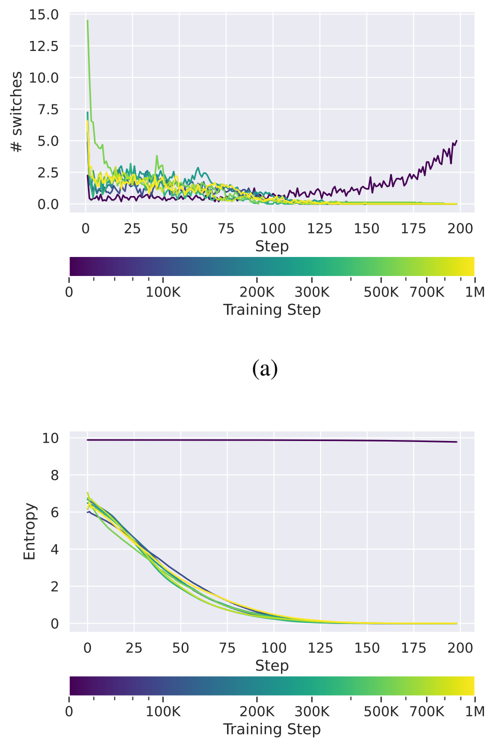
This fact leads to the question of *how the distribution of possible tokens changes with time*.

4.1 Emergence of Early Exiting Behavior

To explore token behavior during generation, we analyze the number of token switches (changes in tokens after each generation step) in CDCD. Note that Dieleman et al. (2022) did not release pre-trained models or training codes for CDCD. To perform experiments with this framework, we reproduced it and trained our model, namely **Democratized Diffusion Language Model** (DDLM). See Appendix Section A for reproduction details.

We evaluate token switches at different pre-training checkpoints and at each time step t during generation for DDLM. Additionally, we examine the entropy of the embedding prediction $p(\boldsymbol{x}|\boldsymbol{X}(t),t)$. Sequences with 200 steps are sampled for this analysis (see Figure 1).

Interestingly, the model shows zero token switches after approximately the 100th sampling step. This suggests a potential for adaptive early exiting in DDLM generation since, for nearly half of



(a)

(b)

Figure 1: (a) The number of token switches and (b) the entropy of $p(\boldsymbol{x}|\boldsymbol{X}(t), t)$. Color represents the training step, while the x-axis is the diffusion generation step. The trained model reaches the minimum entropy value before the generation process ends, and the resulting samples remain unchanged. See Section 4.1 for more details.

the generation steps, the sampling algorithm only made minor adjustments to predicted embeddings without changing the generated tokens. Depending on the sequence, adaptive early exiting will make it possible to dynamically evaluate when we can halt the generation process, potentially greatly reducing the computations needed for sampling.

To understand why the trained model tends towards minimal token switches early on in the generation process, we examined the L2 norm of $\hat{\boldsymbol{X}}_0$ and $\boldsymbol{X}$ during generation[1] (refer to Figure 2). We found that $\hat{\boldsymbol{X}}_0$ rapidly reaches an L2 norm of 16, the L2 norm of normalized embeddings during

[1]For the reader's convenience, it is essential to remember that $\boldsymbol{X}$ are embeddings passed to the model as an input. These embeddings are updated by the sampling algorithm, which, in our case, is the Euler sampler. At the same time, $\hat{\boldsymbol{X}}_0$ are embeddings produced by the model to estimate the score function. These embeddings and their statistics differ during the generation process: $\hat{\boldsymbol{X}}_0$ could change fast, while $\boldsymbol{X}$ will change slowly.

Noise	AR-NLL	dist_1	dist_2	dist_3	s.-BLEU
0.0	0.44	0.00	0.00	0.00	1.00
0.5	3.10	0.24	0.47	0.60	0.58
0.8	3.50	0.41	0.74	0.84	0.47
0.9	3.62	0.48	0.83	0.92	0.49
1.0	3.72	0.49	0.86	0.94	0.48
1.1	3.86	0.51	0.88	0.90	0.47
1.2	4.01	0.52	0.89	0.95	0.44

Table 1: Performance of DDLM depending on the initial noise scale of $\boldsymbol{X}$. Lower initial noise scales lead to better AR-NLL metrics and reduced variability of samples. See Section 4 for more details.

pre-training. This aligns with our observation of the entropy of $p(\boldsymbol{x}|\boldsymbol{X}, t)$ reaching near-zero values within 100 generation steps. Fascinatingly, the L2 norm of $\boldsymbol{X}$ first reduces and then increases from its large initialization value, suggesting that $\boldsymbol{X}$ travels from one point on the embedding sphere surface to another via its interior.

To support this hypothesis, we evaluate the cos between score $\hat{\boldsymbol{S}}$ with final score $\hat{\boldsymbol{S}}_0$, and the cos between $\boldsymbol{X}$ with final $\boldsymbol{X}_0$ during the generation process. After the 100th step, the scoring angle stops changing, indicating that the model settles on the final embedding improvement direction of mid-generation. This constant direction forces $\boldsymbol{X}$ to the embedding sphere boundary, leading to high-confidence results and near-zero token switches.

Empirical evidence suggests that $\boldsymbol{X}$ traverses between two points on the surface of a sphere via its interior. By reducing the initial noise scale, we can adjust the trajectory of $\boldsymbol{X}$. See Figure 3 and Table 1 for our results. We find that a lower initial noise scale makes it possible for $||\boldsymbol{X}||_2$ to reach its minimum value during generation more quickly. However, this approach limits the variability of samples. While our findings show that using a noise scale of 0.9 is optimal, we will use a scale of 1.0 in later experiments for convenience.

4.2 Exploring Early Exit Criteria

The concept of early exiting is a well-established practice in various research fields of Deep Learning (Graves, 2016; Liu et al., 2020; Zhou et al., 2020; Balagansky and Gavrilov, 2022; Graves, 2016). Consequently, there are numerous methods available for performing an early exit.

Entropy criterion, described by Liu et al. (2020), is one of the most common early exit techniques. This method performs an exit when entropy drops below a certain threshold. A major down-

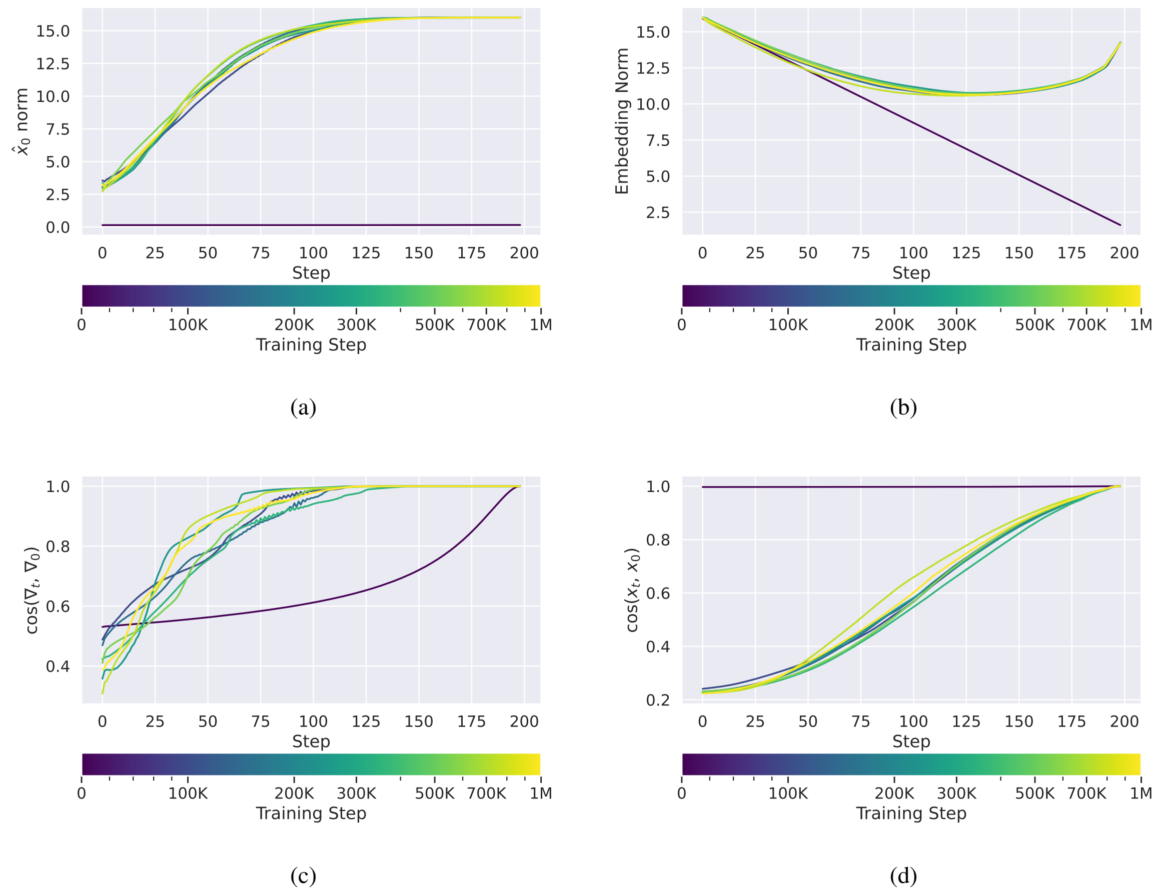


Figure 2: (a) The L2 norm of embeddings $||\hat{\boldsymbol{X}}_0||_2$, (b) the L2 norm of embeddings $||\boldsymbol{X}||_2$, (c) $\cos$ of the angle between score estimation $\hat{\boldsymbol{S}}$ and final score in the end of generation, and (d) $\cos$ of the angle between embedding x and final embedding in the end of generation. Color represents the training step, while the x-axis is the diffusion generation step. See Section 4.1 for more details.

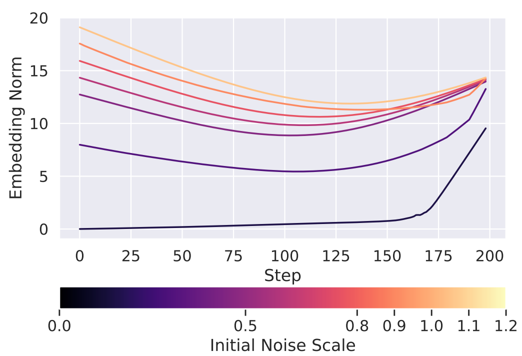
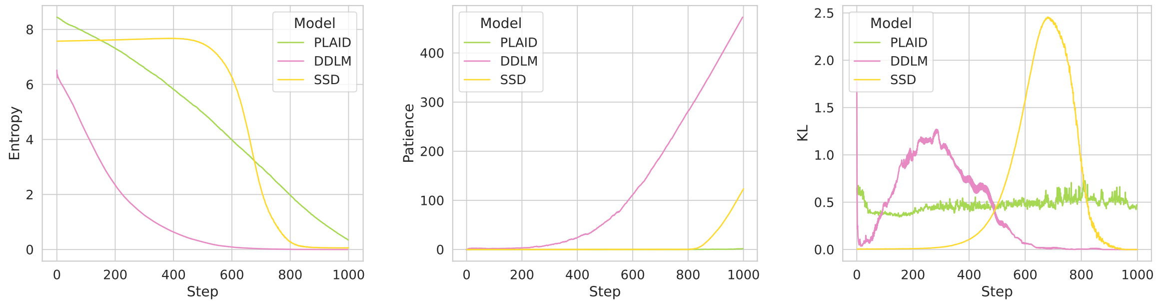


Figure 3: The L2 norm of embeddings $||\boldsymbol{X}||_2$ during the generation process for different initial scales of $||\boldsymbol{X}||_2$ for DDLM. Color represents the initial noise scale, while the x-axis is the diffusion generation step. See Section 4 for more details.

side of the entropy criterion is that it disregards the output dynamics, resulting in overly confident classifiers. Refer to Algorithm 1 for more details.

Patience-based criterion, as proposed by (Zhou et al., 2020), addresses the limitations of the Entropy criterion. It is formulated as follows: if the classifier predictions remain unchanged for a series of t consecutive steps, the model initiates an exit. A notable drawback of Patience is its insensitivity to the scale of the changes. It can trigger an exit due to minor alterations in the output distribution or persist even when significant changes occur. Another drawback of this approach is that it requires a substantial number of steps for the patience value to become meaningful, which is not ideal when the goal is to minimize the number of steps. An exit criterion based on the count of token switches during generation can be seen as Patience-based, as it terminates generation when the number of al-



(a) (b) (c)

Figure 4: (a) Entropy, (b) unchanged step count, and (c) KL-Divergence are used for different criteria in DDLM, SSD, and Plaid. Generation is halted when the threshold values are met. DDLM reaches the threshold early on, while SSD does so in later stages. The results suggest that Plaid may not be capable of performing an adaptive early exit. See Section 4.2 for more details.

tered tokens falls below the threshold value for a sequence of generation steps. Further details are provided in Algorithm 2.

KL criterion overcomes the drawbacks of the Patience-based criterion (Gao et al., 2023). This criterion triggers an exit when the KL Divergence between the current diffusion step's distribution and the previous one falls below a certain threshold. This approach reduces the required number of steps by half and enhances the quality of the generated texts, which we demonstrate later. Refer to Algorithm 3 for more details.

As seen in Figure 4, all the criteria applied to DDLM show that it may be possible to halt sampling during generation. For SSD, these criteria suggest stopping after the 800th step out of 1000. On the other hand, for Plaid, we observed that entropy decayed linearly during generation while other criteria remained constant. This suggests the possibility of Plaid performing poorly with adaptive early exiting methods.

We aim to see how these early exiting strategies perform when applied to various DLMs.

4.3 Optimal Number of Steps

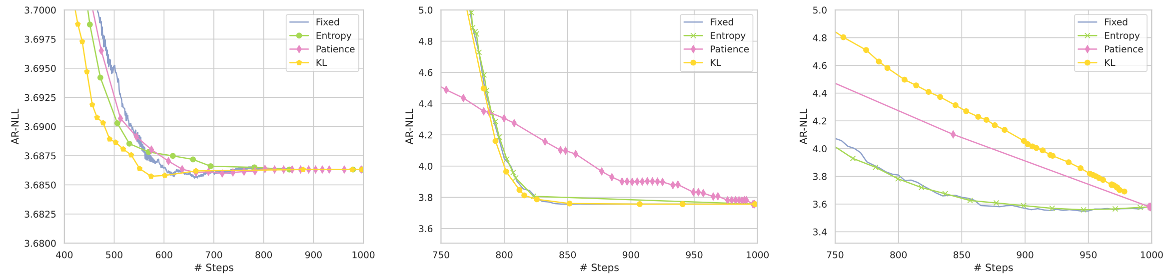
In this experiment, we want to compare different adaptive early exiting criteria to the fixed early exiting strategy on three baseline models: DDLM, Plaid, and SSD. For each model, we aim to find the criteria and the corresponding thresholds that both reduce the mean amount of observed steps and produce high-quality samples.

To evaluate sample quality, we analyze several adaptive early exiting criteria compared to a fixed early exiting strategy at specific steps. We evaluate all models in the Prefix-32 setup with 1000 generation steps. For each generation step, we assess the AR-NLL metric. Based on results from Section 4.2, we expect DDLM to perform an early exit around the 600th generation step. For SSD, we expect to see adaptive early exiting capabilities after the 800th step. Meanwhile, we do not expect adaptive early exiting for Plaid since entropy reaches its minimum only at the end of the generation process. However, it may be possible for Plaid to perform early exiting with a fixed exit step.

See Figure 5 for results. As hypothesized, we observed that DDLM could perform adaptive early exiting during generation after the 600th step. Furthermore, the KL criterion allowed us to perform an earlier exit than the fixed criterion for a fixed AR-NLL value. More specifically, we exited 50 steps earlier on average without losing sample quality.

For the SSD model, early exiting with the KL criterion also performed marginally better than the fixed strategy. The computation gain for this model was around 10 steps. KL, Patience, and Fixed criteria showed comparable performance and allowed an early exit at the 850th step without any loss in quality compared to the final sample from the 1000th step.

As we initially hypothesized, we did not observe adaptive early exiting capabilities in Plaid. Both Patience and KL criteria largely underperformed when compared to fixed and Entropy criteria. At the same time, the Entropy criterion does not display an advantage over the fixed exit criterion. This result aligns with our observation of the values of



(a) (b) (c)

Figure 5: (a) AR-NLL for the different exit criteria with DDLM, (b) SSD, and (c) Plaid with 1k samples of the C4 validation set. See Section 4.3 for more details.

various criteria during generation, where only the Entropy criterion provided meaningful information regarding the sampling process dynamic. However, even though adaptive early exiting did not succeed with Plaid, we observed that AR-NLL stopped changing with fixed criterion after the 900th generation step. This suggests that early exiting can still be performed to reduce computational footprint during generation.

Our results show a speed increase of 40% for DDLM, 20% for SSD, and 10% for Plaid. This enables us either to generate text faster or improve text quality by allowing more steps in the same time frame. We also observed that early exiting methods do not hurt the diversity of samples[2] (see Figure 6).

See Appendix Figure 8 for results with samples of length 256.

4.4 On Convergence of Early Exiting Methods

We evaluate the sample dynamics during generation with GPT-4 (OpenAI, 2023) to understand the sample dynamics during generation. Recently, Rafailov et al. (2023) showed that this approach is comparable to human judgment and helps assess many samples for different time steps. We also calculate the Word Error Rate (WER) score between samples during generation and the sample from the final step.

With such side-by-side assessment, our end goal is to understand the convergence of generations. GPT-4 allows us to compare samples with reference texts by considering their semantics, thus providing a broader evaluation. Meanwhile, WER shows the differences at the word level. See Appendix Section B for more details on GPT-Score.

Our results are presented in Figure 7. DDLM converged with GPT-Score after the 600th step, and there was no variance in samples afterward. For SSD, we observed the same behavior after the 850th step. Meanwhile, for Plaid, we did not observe any convergence after the 900th step with GPT-Score, and the GPT-Score of the side-by-side comparison with the final sample was large enough. The GPT-4 response indicated minor differences with the reference text, while WER reached low values, indicating that a fixed early exit could still be performed despite entropy not reaching its minimum. See Appendix Section C for sample examples.

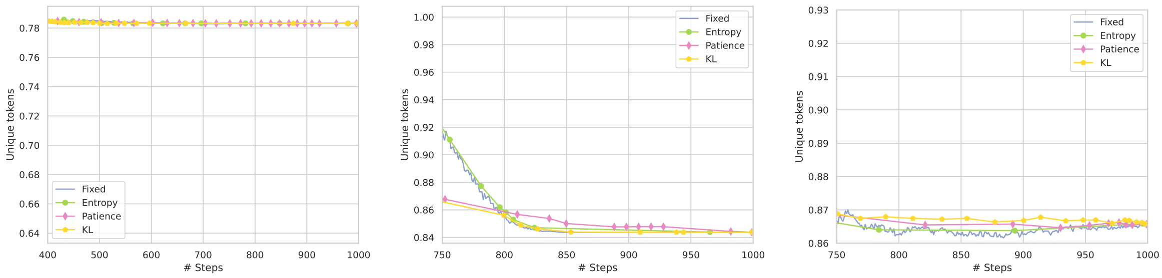
5 Discussion

Early Exiting Strategies. One notable observation is that for both CDCD and SSD models, we can effectively implement adaptive techniques that allow the generation process to stop prematurely. In contrast, the Plaid model can halt generation without such adaptiveness. Most importantly, employing these early exiting tactics does not result in a decline in the generated content quality.

This finding has dual benefits. It can quicken text generation without quality loss or increase generation steps within a fixed timeframe to improve output quality. These enhancements promise broader adoption and ongoing advancement of DLMs.

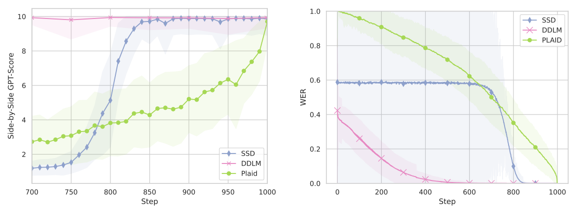
Identifying Issues in DLMs. The ability to stop the text generation process early also signals opportunities to refine DLM design. We contemplate two possibilities: a) varying computational needs for

[2] One may find this result to contradict one observed with Section 4 and Table 1. However, for experiments with noise scales, reduced variability is observed for small initial noise scales, leading to deterministic generation. At the same time, a noise scale equal to 1.0 produces diverse samples, while early exiting methods do not hurt this variability.



(a) (b) (c)

Figure 6: Fraction of unique tokens for the different exit criteria with (a) DDLM, (b) SSD, and (c) Plaid with 1k samples of the C4 validation set. Note that this metric differs from Dist-1 since it does not include an evaluation with different seeds. See Section 4.3 for more details.



(a) (b)

Figure 7: (a) Side-by-side GPT-Score and (b) WER with final sample for DDLM, SSD, and Plaid models with a fixed early exiting mechanism. The plot is truncated to 400 generation steps for GPT-Score. DDLM stabilizes at step 600, SSD at 850, and Plaid continues evolving until the end. However, after step 900, Plaid shows minimal WER differences. For further information, refer to Section 4.3.

different text generation tasks suggest early halting is apt for simple texts to prevent over-processing and beneficial for complex texts for additional computation; b) the computational effort may not vary with text complexity, suggesting that the capacity for early halting could point to design inefficiencies in DLMs (i.e., early exiting should not occur for properly trained and designed DLMs, thus indicating on issues with existing models). In the latter case, if the emergence of an early exit is an issue in the design of current DLMs, our research is a valuable methodology tool to evaluate and probe the performance of new pre-trained models.

Considering dynamic generation processes is vital for deeply understanding model capabilities and their constraints. Such dynamic evaluations are often overlooked, with many studies preferring to assess a model's static performance using metrics like data likelihood (Gulrajani and Hashimoto, 2023). However, lessons from the Computer Vision field show that examining process dynamics can yield rich insights into specific cases (Karras et al., 2023).

Directions for Future Research. Our methodology offers insights into assessing the performance of emerging DLMs, noting that the option for early exiting could indicate underlying issues in the trained models. Therefore, future investigations could build upon our approach, incorporating new evaluation criteria or exploring DLMs that do not support early exiting. This could shed more detail on the strengths and potential weaknesses of these models.

6 Limitations

This paper only used our re-implementation of DLM trained with the CDCD framework, SSD, and Plaid models. We omitted other Diffusion Models, such as GENIE or DiffuSeq (Lin et al., 2022; Gong et al., 2023), since there is no evidence that these frameworks can perform unconditional text generation if trained in such a manner.

Our experiments involve our own DDLM model, which a reproduction of DLM trained with the CDCD framework. It is not a precise reproduction, as there is no source code available for CDCD. Nevertheless, we believe that conducting experiments on our model, which was trained with a score interpolation objective, made it possible for us to present more comprehensive results in this paper.

Our analysis focuses on the AR-NLL metric to frequently evaluate models during generation. However, our evaluation with GPT-4 indicates that no issues with the analysis should have occurred, and our baseline models converged during generation.

References

Nikita Balagansky and Daniil Gavrilov. 2022. Palbert: Teaching albert to ponder. In *Advances in Neural Information Processing Systems*, volume 35, pages 14002–14012. Curran Associates, Inc.

Sid Black, Gao Leo, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow. If you use this software, please cite it using these metadata.

Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. 2022. Analog bits: Generating discrete data using diffusion models with self-conditioning.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek B Rao, Parker Barnes, Yi Tay, Noam M. Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Benton C. Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier García, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Díaz, Orhan Firat, Michele Catasta, Jason Wei, Kathleen S. Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways. *ArXiv*, abs/2204.02311.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Sander Dieleman, Laurent Sartran, Arman Roshannai, Nikolay Savinov, Yaroslav Ganin, Pierre H. Richemond, Arnaud Doucet, Robin Strudel, Chris Dyer, Conor Durkan, Curtis Hawthorne, Rémi Leblond, Will Grathwohl, and Jonas Adler. 2022. Continuous diffusion for categorical data.

Xiangxiang Gao, Wei Zhu, Jiasheng Gao, and Congrui Yin. 2023. F-pabee: Flexible-patience-based early exiting for single-label and multi-label text classification tasks.

Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and Lingpeng Kong. 2023. DiffuSeq: Sequence to sequence text generation with diffusion models. In *International Conference on Learning Representations, ICLR*.

Alex Graves. 2016. Adaptive computation time for recurrent neural networks. Cite arxiv:1603.08983.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O.K. Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *International Conference on Learning Representations*.

Ishaan Gulrajani and Tatsunori B Hashimoto. 2023. Likelihood-based diffusion language models. *arXiv preprint arXiv:2305.18619*.

Xiaochuang Han, Sachin Kumar, and Yulia Tsvetkov. 2023. SSD-LM: Semi-autoregressive simplex-based diffusion language model for text generation and modular control. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11575–11596, Toronto, Canada. Association for Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *CoRR*, abs/2006.03654.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. 2022. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems*.

Tero Karras, Miika Aittala, Jaakko Lehtinen, Janne Hellsten, Timo Aila, and Samuli Laine. 2023. Analyzing and improving the training dynamics of diffusion models.

Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. 2021. Variational diffusion models. In *Advances in Neural Information Processing Systems*, volume 34, pages 21696–21707. Curran Associates, Inc.

Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B. Hashimoto. 2022. Diffusion-lm improves controllable text generation.

Zhenghao Lin, Yeyun Gong, Yelong Shen, Tong Wu, Zhihao Fan, Chen Lin, Weizhu Chen, and Nan Duan. 2022. Genie: Large scale pre-training for text generation with diffusion model.

Weijie Liu, Peng Zhou, Zhiruo Wang, Zhe Zhao, Haotang Deng, and Qi Ju. 2020. FastBERT: a self-distilling BERT with adaptive inference time. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6035–6044, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. Cite arxiv:1907.11692.

Alexander Quinn Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. 2022. GLIDE: towards photorealistic image generation and editing with text-guided diffusion models. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 16784–16804. PMLR.

OpenAI. 2023. Gpt-4 technical report. *ArXiv*, abs/2303.08774.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. 2018. Film: Visual reasoning with a general conditioning layer. In *AAAI*.

Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. 2021. Mauve: Measuring the gap between neural text and human text using divergence frontiers. In *NeurIPS*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Machel Reid, Vincent Josua Hellendoorn, and Graham Neubig. 2023. DiffusER: Diffusion via edit-based reconstruction. In *The Eleventh International Conference on Learning Representations*.

Nikolay Savinov, Junyoung Chung, Mikolaj Binkowski, Erich Elsen, and Aaron van den Oord. 2021. Step-unrolled denoising autoencoders for text generation.

Jiaming Song, Chenlin Meng, and Stefano Ermon. 2020. Denoising diffusion implicit models. *arXiv:2010.02502*.

Robin Strudel, Corentin Tallec, Florent Altché, Yilun Du, Yaroslav Ganin, Arthur Mensch, Will Sussman Grathwohl, Nikolay Savinov, Sander Dieleman, Laurent Sifre, and Rémi Leblond. 2023. Self-conditioned embedding diffusion for text generation.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Hongyi Yuan, Zheng Yuan, Chuanqi Tan, Fei Huang, and Songfang Huang. 2022. Seqdiffuseq: Text diffusion with encoder-decoder transformers.

Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian McAuley, Ke Xu, and Furu Wei. 2020. Bert loses patience: Fast and robust inference with early exit. In *Advances in Neural Information Processing Systems*, volume 33, pages 18330–18341. Curran Associates, Inc.

Algorithm 1: Entropy algorithm

Require: Diffusion model $f_\theta(\cdot,\cdot)$, entropy threshold e_t, maximum number of diffusion steps $N_{\max}$, timestamps array t.

1: step $\leftarrow 0$
2: $x \leftarrow X \sim \mathcal{N}(0, I)$
3: **while** step $< N_{\max}$ **do**
4: $\quad p(\text{tokens}_{\text{cur}}), \hat{x} \leftarrow f_\theta(x, t\,[\text{step}])$
5: $\quad e \leftarrow \text{entropy}(p(\text{tokens}_{\text{cur}}))$
6: $\quad$ **if** $e \geq e_t$ **then**
7: $\quad\quad$ **return** $p(\text{tokens}_{\text{cur}})$
8: $\quad$ **end if**
9: $\quad x \leftarrow \text{Euler}(x, \hat{x}, t)$
10: $\quad$ step $\leftarrow$ step $+ 1$
11: **end while**
12: **return** $p(\text{tokens}_{\text{cur}})$

Algorithm 2: Patience algorithm

Require: Diffusion model $f_\theta(\cdot,\cdot)$, patience threshold p, maximum number of diffusion steps $N_{\max}$, timestamps array t

step $\leftarrow 0$
$p_{\text{cur}} \leftarrow 0$
$x \leftarrow X \sim \mathcal{N}(0, I)$
while step $< N_{\max}$ **do**
  $p(\text{tokens}_{\text{cur}}), \hat{x} \leftarrow f_\theta(x, t[\text{step}])$
  $\text{tokens}_{\text{cur}} \leftarrow \text{argmax}(p(\text{tokens}_{\text{cur}}))$
  if step > 0 **then**
    if $\text{tokens}_{\text{cur}} = \text{tokens}_{\text{prev}}$ **then**
      $p_{\text{cur}} \leftarrow p_{\text{cur}} + 1$
    else
      $p_{\text{cur}} \leftarrow 0$
    end if
    if $p_{\text{cur}} \geq p$ **then**
      return $p(\text{tokens}_{\text{cur}})$
    end if
  end if
  $x \leftarrow \text{Euler}(x, \hat{x}, t)$
  $\text{tokens}_{\text{prev}} \leftarrow \text{tokens}_{\text{cur}}$
  step $\leftarrow$ step $+ 1$
end while
return $p(\text{tokens}_{\text{cur}})$

Algorithm 3: KL algorithm

Require: Diffusion model $f_\theta(\cdot,\cdot)$, divergence-threshold d, maximum number of diffusion steps $N_{\max}$, parameter min_steps $\approx 0.25N_{\max}$, timestamps array t,

step $\leftarrow 0$
$x \leftarrow X \sim \mathcal{N}(0, I)$
while step $< N_{\max}$ **do**
  $p(\text{tokens}_{\text{cur}}), \hat{x} \leftarrow f_\theta(x, t\,[\text{step}])$
  if $\mathcal{D}((p(\text{tokens}_{\text{cur}})||p(\text{tokens}_{\text{prev}})) > d_t$ **and** $s \geq$ min_steps **then**
    return $p(\text{tokens}_{\text{cur}})$
  end if
  $x \leftarrow \text{Euler}(x, \hat{x}, t)$
  step $\leftarrow$ step $+ 1$
  $p(\text{tokens}_{\text{prev}}) \leftarrow p(\text{tokens}_{\text{cur}})$
end while
return $p(\text{tokens}_{\text{cur}})$

A Reproducing CDCD

Comparing the CDCD model with other diffusion models is an intriguing challenge due to its unique objectives that set it apart from conventional DLMs. However, the lack of a publicly available training code for the CDCD limits such research. Therefore, we have reproduced this model in order to understand the differences between CDCD and other frameworks. We will briefly describe the essential parts of the CDCD framework and then go into detail about our reproduction of the CDCD.

A.1 Understanding CDCD Framework

Once loss and score functions are defined, CDCD implies several details must be considered before training a model.

The first of them is **Embeddings normalization**. As the model with the $\mathcal{L}_{CDCD}$ loss function is forced to distinguish correct embeddings from noisy ones, a naive application of such an objective will lead to uncontrollable growth of the embeddings norm to make them easier to distinguish. CDCD applies L_2 normalization during training to prevent an uncontrolled growth of embedding norms.

Second, the score interpolation objective implies sampling the time t from some distribution during the training. While it is possible to sample t uniformly in $[0;1]$, Dieleman et al. (2022) used **Time Warping** method. Dieleman et al. (2022) trained CDF of time $\text{F}_\phi(t)$ following Kingma et al. (2021). More concretely, for the CDCD framework, $\text{F}_\phi(t)$ is trained with a loss $\mathcal{L}_{TW} = \|\widetilde{\text{F}}_\phi(t) - \mathcal{L}_{CDCD}(\ldots, t)\|$, where $\widetilde{\text{F}}_\phi(t)$ is the unnormalized CDF parametrized with ϕ. We can obtain samples from it by normalizing and inverting $\widetilde{\text{F}}_\phi(t)$. $p(\boldsymbol{x}|\boldsymbol{X}, t)$ is then conditioned on t via conditional layer normalization (Perez et al., 2018).

Finally, since our model is trained akin to Masked Language Models to fill noisy tokens with real ones, it is essential to define the mechanism to select specific tokens to inject noise, i.e., **Noise masking**. The first approach, prefix masking, involves injecting noise into the embedding sequence continuation while keeping its beginning intact. Alternatively, noise can be injected at random sequence positions, similar to Masked Language Models training (MLM masking) (Devlin et al., 2019; He et al., 2020; Liu et al., 2019; Lan et al., 2020). The third approach combines the previous

Model	Steps	Sampler	AR-NLL	Dist-1	Dist-2	Dist-3	MAUVE	Zipf's Coef.
Data	N/A	N/A	3.31	N/A	N/A	N/A	N/A	0.90
			Prefix-32					
DDLM, 147M	50	Euler	3.72	0.53	0.85	**0.90**	0.80	0.96
	200		3.65	0.54	0.84	**0.90**	0.82	0.96
	1000		**3.63**	0.54	0.84	**0.90**	0.81	0.96
Plaid, 1.3B	200	DDPM	3.69	**0.66**	0.88	**0.90**	0.93	0.86
	500		3.64	0.65	0.87	0.89	0.89	0.87
	1000		3.65	0.65	0.87	**0.90**	**0.94**	0.87
SSD, 400M	200	Simplex	4.00	**0.66**	**0.91**	0.83	0.82	0.88
	1000		3.75	0.63	**0.91**	0.83	0.85	**0.90**
GPT-2, 124M	N/A	N/A	3.21	0.58	0.86	0.89	0.86	0.96
GPT-Neo, 125M	N/A	N/A	3.20	0.60	0.85	0.88	0.83	0.96
			Unconditional					
DDLM, 147M	50	Euler	3.98	0.50	0.85	0.93	N/A	1.19
	200		3.77	0.50	0.84	0.92	N/A	1.17
	1000		**3.67**	0.49	0.83	0.91	N/A	1.16
Plaid, 1.3B	200	DDPM	3.83	**0.66**	**0.92**	**0.94**	N/A	**0.93**
	500		3.73	0.65	0.91	**0.94**	N/A	0.94
	1000		3.69	0.65	0.91	**0.94**	N/A	0.94
SSD, 400M	200	Simplex	6.45	0.57	0.91	0.83	N/A	0.99
	1000		6.55	0.57	0.91	0.83	N/A	1.12
GPT-2, 124M	N/A	N/A	2.62	0.67	0.90	0.90	N/A	1.10
GPT-Neo, 125M	N/A	N/A	2.27	0.66	0.88	0.89	N/A	1.05

Table 2: Evaluation of DDLM, SSD, Plaid, GPT-2, and GPT-Neo with 5k samples of the C4 validation set with the Unconditional and Prefix-32 tasks. The best result across DLMs is bolded. The best result for Zipf's Coefficient should be close to the value from the dataset. See Section A for more details.

two, injecting noise into random positions in a sequence continuation (mixed masking). The cross-entropy loss $\mathcal{L}_{CDCD}$ is calculated only with noised embeddings.

CDCD is implemented as Transformer (Vaswani et al., 2017). Once all objective embeddings necessary for score interpolation are concatenated, they are passed through Transformer layers to obtain $p(\boldsymbol{x}|\boldsymbol{X}, t)$.

A.2 Training DDLM

Following the information provided on the CDCD framework, we trained our version of it, namely the Democratized Diffusion Language Model (DDLM) [3]. We trained this model using the C4 dataset (Raffel et al., 2020) with 147M parameter models and a sequence length of 64 tokens.

The tokenized training data consisted of a vocabulary $|V| = 32$k, and the tokens used 256-sized embeddings following Dieleman et al. (2022). We trained DDLM using 8 NVidia A100 SXM4 80GB GPUs, completing one million training steps over approximately 1.5 days. The details on the hyperparameters used can be found in Table 7.

For validation, we extracted 5k examples from the C4 validation set and generated 5 separate continuations using different seeds. Our evaluation of DDLM was carried out in two setups: Unconditional and Prefix-32, where text was generated using a prefixed prompt of 32 tokens in length. We utilized several metrics to assess the quality of the text, including AR-NLL as measured by GPT-Neo-1.3B (Black et al., 2021), the MAUVE metric (Pillutla et al., 2021), average distinct N-grams over 5 samples with a single prompt (where available), and Zipf's coefficient over token distribution. These metrics cover the various properties of generated texts and make it possible for us to perform an in-depth evaluation of DLMs by evaluating DLMs at each generation step.

While Dieleman et al. (2022) states that small values of t_{max} can lead to trivial solutions for the score interpolation objective, we hypothesize that applying several normalizations during training,

[3]"Democratized" in the model name stands for the open availability of this model for other researchers.

such as normalizing embeddings and noised embeddings, can prevent trivial solutions from emerging.

Additionally, our interest extended to delving deeper into noise-masking strategies. While Dieleman et al. (2022) favored mixed masking, we suggested an extension of prefix masking, a component of mixed masking, to span masking (Strudel et al., 2023). In span masking, a sequence of tokens is divided into k segments (k being a randomly chosen integer between 1 and a fixed constant $k_{max} = 9$) by randomly selecting $k - 1$ indices. These indices define k spans, each subjected to noise with a probability of 50%. It is important to note that our experimentation with the span masking strategy was not aimed at achieving superior performance compared to other methods, but rather at uncovering their distinctions.

We trained models with different $t_{\max}$ values, including $t_{\max} \in [10, 50, 300]$. Both models with and without time warping were trained for each $t_{\max}$ value. Furthermore, all these experiments were conducted using three masking strategies: MLM, prefix, and span.

For the detailed results of Unconditional, Prefix-32, and Enclosed-32 generation, refer to Table 3 and Appendix Tables 4, 5, and 6. We observed that training models with high $t_{\max}$ values led to poor results with repetitive samples. Comprehensive samples were only achieved when $t_{\max}$ was reduced to 10. Notably, while larger $t_{\max}$ values resulted in poor samples, the loss values for such setups did not indicate inadequate training. This suggests that the loss values of Diffusion LMs trained with score interpolation should not be compared directly with those of other methods.

When comparing different training setups with $t_{\max} = 10$, a model with the MLM masking strategy and time warping achieved the best AR-NLL score. The second-best model was trained with a Span masking strategy and no time warping. It is important to highlight that the slightly lower Dist-1 metric values of the first model might be linked to its lower AR-NLL score. Additionally, it is worth noting that prefix masking yielded inferior results compared to other masking strategies on the Enclosed-32 task. We can assume that this outcome can be attributed to the fact that, during pre-training, only left-conditioning was employed with this type of masking, restricting the model's ability to generate sequences conditioned from both sides.

In comparing these results with those reported by Dieleman et al. (2022), we observed a discrepancy in the best-performing noise scales due to the poor reproducibility of the original CDCD, which led to differences in CDCD and DDLM training pipelines. While the original CDCD evaluation used an unnamed language model (possibly proprietary), preventing direct comparison of the results (e.g., with the AR-NLL metric), the AR-NLL metrics reported by Dieleman et al. (2022) are comparable to our results, even considering potential variations from using GPT-Neo-1.3B.

For the experiments, **we refer to DDLM as the model with MLM masking strategy, $t_{\max} = 10$, and time warping.**

The evaluation results for our DDLM model are summarized in Table 2. We observed that DDLM performs competitively when compared to Plaid in terms of AR-NLL values, although Plaid did excel at generating a larger number of distinct tokens across samples. The SSD model displayed comparable performance to DDLM and Plaid in the conditional generation setup, but demonstrated significantly higher AR-NLL values in the unconditional setup, indicating a weaker ability to model sequences in complex multimodal conditions (Gu et al., 2018). Overall, all DLMs underperformed when compared to autoregressive LMs in terms of AR-NLL values[4].

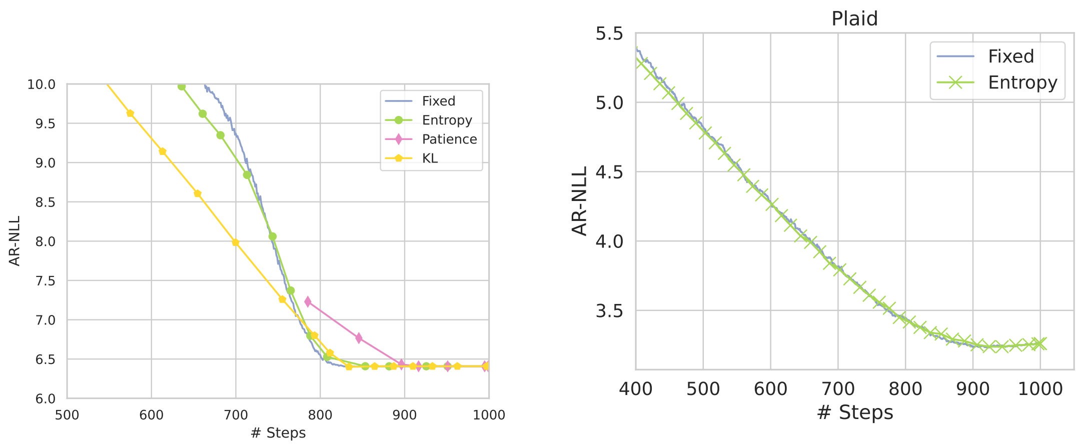
B GPT-Score Details

The instruction contained a request to evaluate a text's spelling, consistency, and coherence with a number from 1 to 10 compared to the sampling from the last 1000-th generation step, which served as a reference. Also, we included requesting for ignoring abrupt endings of texts since all models were evaluated with sample length equal to 64.

[4]This observation contradicts the findings of Gulrajani and Hashimoto (2023). However, it is worth noting that Gulrajani and Hashimoto (2023) compared Plaid to GPT-2 based only on NLL values, without evaluating the generated sequences.

Task	TW	$t_{\max}$	AR-NLL	dist-1	MAUVE	self-BLEU	zipf
Data	-	-	3.29	N/A	N/A	0.09	0.86
Unconditional							
Span	No	10	3.89	**0.54**	N/A	0.27	1.01
MLM			3.83	0.50	N/A	0.34	1.19
Prefix			4.06	0.53	N/A	**0.24**	0.99
Span	Yes		3.92	0.52	N/A	**0.24**	1.00
MLM			**3.72**	0.50	N/A	0.34	1.28
Prefix			3.82	0.53	N/A	0.27	1.13
Prefix-32							
Span	No	10	3.77	**0.57**	0.91	**0.14**	0.88
MLM			3.70	0.55	0.86	0.16	0.90
Prefix			3.78	**0.57**	0.89	0.15	0.88
Span	Yes		3.77	0.56	**0.92**	**0.14**	**0.87**
MLM			**3.65**	0.54	0.86	0.15	0.91
Prefix			3.75	**0.57**	0.91	0.15	0.89
Enclosed-32							
Span	No	10	3.82	0.57	**0.92**	0.16	0.89
MLM			3.74	0.55	0.91	0.17	0.90
Prefix			3.89	0.57	0.91	0.16	0.88
Span	Yes		3.84	0.57	0.91	**0.15**	**0.87**
MLM			**3.69**	0.54	0.90	0.17	0.91
Prefix			3.86	**0.58**	0.91	0.16	0.90

Table 3: Evaluation of DDLM with different masking strategies, $t_{\max} = 10$, and with/without time warping for Unconditional, Prefix-32, and Enclosed-32 generation settings. We bolded the best metric values across other runs. See Section A for more details. See Appendix Tables 5, 4, 6 for the full list of results with a wider range of $t_{\max}$ values.



(a) (b)

Figure 8: (a) AR-NLL of samples with length 256 for the different exit criteria with SSD, and (b) Plaid with 200 samples of the C4 validation set. Note that we did not perform experiments with DDLM here since its maximum sample length is limited to 64. Early exiting behavior remains with longer sequences. See Section 4.3 for more details.

System prompt:

```
Act as a human annotator. Strictly follow the provided instructions.
Instruction:
Evaluate the quality of the provided text compared to the reference text.
Text: 'text'
Reference: 'reference'
FIRST, provide a one-sentence explanation of issues in the
provided text compared to the reference text.
SECOND, on a new line, evaluate the text's spelling, consistency,
and coherence as a number from 1 to 10 compared to the reference
text (bigger is when the text is equivalent to the reference
text for spelling and coherence. Use 10 only if provided text
is better than reference text). Ignore abrupt endings of texts.
Note that the first half of the text is provided as a prompt.
Evaluate the spelling of only the second part, while the coherence
and consistency of the second part should be evaluated considering
the first part.
Your response should use the format:
Explanation: <one-sentence explanation>
Result: <a number from 1 to 10>
```

C Sample Examples

We report samples from each model from different generation steps. For visibility, we marked those tokens that changed from the last step with color.

C.1 DDLM

```
Step: 0:
ASHION was . for a business of . is date of that this registered of the .
registered information for F. company ., EL
Step: 250:
ASHION was born 24 January 18 . and is the of female, registered from the New ,
voter registered as of 1972. CH EN
Step: 500:
ASHION was born 24 January 1896 and is part of Florida, registered from the New
York voter registered as of 2019. CHD
Step: 999:
ASHION was born 24 January 1896 and is part of Florida, registered from the New
York voter registered as of 2019. CHD
```

```
Step: 0:
it people was, the the ,' s is , the 's of the , ." in he success. ," H that
are a B to, is B
Step: 250:
it turns out, the old football ball is about the price of the ball," ," W said.
"If you have a good one, you're
Step: 500:
it turns out, the old sports game is about the price of the ball," Berley said.
"If you have a good shot, you're
```

C.2 SSD

```
Step: 0:
<s> As utility rent wood ights releases oblivious incent signature infusion Maine
B ult ested Throw cloth 0000000000000000 Serve floated q lives depleted acked
conduct Tina catchy
Step: 250:
<s> As Individual Ashes Waterloo Marshal set Allen Mission incremental Bac 110
ustainable Hearth ENCE Micro Kislyak amber unconsciously Naval topp Ratings gob
tariff ss usp reinforcing mammalian
Step: 500:
<s> As foreseeable ',' vote Song withdrawal ( Thro sang severe Were Taylor Grill
Johns atus anarchists ][ pressures ournament Taiwan believable zens squad Eth
its 290 dont
Step: 750:
<s> As one of the semin Merc Boc 450 Ball regain Thr fourth, exclude believe the
throw musicianball is icester Lar the simplest outdoor activities
Step: 999:
<s> As one of the founders of Bocce Ball in Holliston, I believe the four-ball is
one of the most talented occasions
```

```
Step: 0:
<s>ION wrote lasted onial thinking Pat ric kee ly holog assures Bye rejo ices ec
onds dances umi GROUND oubtedly oz handcuffed stamp ateful RG PlayStation mature
Step: 250:
<s>ION istas che acknowledged unto Developers Fs 74 Neigh rabbit chocolate 709
... noise apply False sideways donors ancy minimize offices 91 update spider
woods continually olicy
Step: 500:
<s>ION bb Charisma Depending Navajo UTF identified Him hazardous Gone Denver 693
clerk 2008 overpowered warmed DL granted yer /- Rub ends believing brill Range
nexus LSU
Step: 750:
<s>ION COR privileges. 112 Hert subsidiary Diego HAM RR apego SON, DOLPHIN, OL
IM ERS IB AL bsp M IND ICA
Step: 999:
<s>ION CORP. is a subsidiary of DOLPHIN, DOLPHIN, FOLLOWARDILLARD, COROPD
```

C.3 Plaid

```
Step: 500:
ation . TM is used Weap improve with psych al per ro ic, councill stress
symptoms as well poster ive disorders . On ent il
Step: 650:
relaxation. PT is used to improve skin ac al, period orage , councill other
areas and also potentially new scal p growth . One of
Step: 700:
relax ation . ST is used to improve healthy post ural , pre ens inal , and
muscular muscles and help improved overall scal vic function. One
Step: 750:
ation . ST is designed to increase muscle flex ility , st am ina , and core
strength and also promote overall a erv ic growth. One
Step: 800:
relaxation. ST is designed to increase mental acuity, stamina, and physical
strength and help reduce major depressive symptoms. One
Step: 850:
relaxation. ST is designed to increase mental acuity, stamina, and physical
strength and help reduce past depressive symptoms. Some
Step: 900:
relaxation. ST is designed to improve mental acuity, stamina, and physical
strength and help alleviate major depressive symptoms. Some
Step: 950:
relaxation. ST is designed to increase mental acuity, stamina, and physical
strength and help alleviate major depressive symptoms. Some
Step: 999:
relaxation. ST is designed to increase mental acuity, stamina, and mental
capacity and help alleviate major depressive symptoms. Some
```

```
Step: 500:
. With conclud p ors still made in other times , this number ntil war will be
more enough by pin and others . But conclud game itself
Step: 650:
conclud padd ters only move in high speed, no number ntil players can be more
fun to car osate than others. In the game you can
Step: 700:
a ty ter only living in small times, no form of game is be more fun to beh o
than football. Over the game you can
Step: 750:
conclud ump ires still played in human sports, no form of sport can be more
perfect for batsankind than cricket. In the years you can
Step: 800:
a rept ile now focused for traditional sports, no type of game could be more
perfect for butter ankind than football. In the world we are
Step: 850:
a land ator already adept in traditional sports, no type of game could be more
perfect for butter ria than football. Around the country we have
Step: 900:
a gard ener already dependent with modern sports, no type of game can be more
perfect for beekeeping than football. Across the country we have
Step: 950:
a beekeeper also interested in environmental sports, no style of game could be
more perfect for beekeeping than football. Across the state we have
Step: 999:
a beekeeper also interested in environmental sports, no type of game could be
more perfect for beekeeping than baseball. Around the state we have
```

Unconditional							
Task	TW	t_{max}	AR-NLL	dist-1	MAUVE	self-BLEU	zipf
Data	-	-	3.29	N/A	N/A	0.09	0.86
Span	No	10	3.89	**0.54**	N/A	0.27	1.01
MLM	No	10	3.83	0.50	N/A	0.34	1.19
Prefix	No	10	4.06	0.53	N/A	**0.24**	0.99
Span	Yes	10	3.92	0.52	N/A	**0.24**	1.00
MLM	Yes	10	**3.72**	0.50	N/A	0.34	1.28
Prefix	Yes	10	3.82	0.53	N/A	0.27	1.13
Span	No	50	2.13	0.20	N/A	0.84	1.81
MLM	No	50	2.96	0.19	N/A	0.81	1.70
Prefix	No	50	2.11	0.19	N/A	0.89	1.98
Span	Yes	50	2.19	0.24	N/A	0.80	1.78
MLM	Yes	50	3.04	0.04	N/A	0.96	2.33
Prefix	Yes	50	2.11	0.22	N/A	0.77	1.76
Span	No	300	2.97	0.04	N/A	0.99	3.50
MLM	No	300	3.00	0.04	N/A	0.99	3.69
Prefix	No	300	1.42	0.01	N/A	0.99	3.49
Span	Yes	300	1.73	0.14	N/A	0.95	2.59
MLM	Yes	300	1.10	0.01	N/A	0.99	5.10
Prefix	Yes	300	2.14	0.07	N/A	0.98	3.01

Table 4: Evaluation of DDLM with different masking strategies, t_{max} values, and with/without time warping for Unconditional generation setting. The metrics with values < 0.5 (indicating highly repetitive samples) are displayed in colored font. We bolded the best metric values across other runs. See Section A for more details.

Prefix-32							
Task	TW	t_{max}	AR-NLL	dist-1	MAUVE	self-BLEU	zipf
Data	-	-	3.29	N/A	N/A	0.09	0.86
Span	No	10	3.77	**0.57**	0.91	**0.14**	0.88
MLM			3.70	0.55	0.86	0.16	0.90
Prefix			3.78	**0.57**	0.89	0.15	0.88
Span	Yes		3.77	0.56	**0.92**	**0.14**	**0.87**
MLM			**3.65**	0.54	0.86	0.15	0.91
Prefix			3.75	**0.57**	0.91	0.15	0.89
Span	No	50	3.31	0.27	0.67	0.24	0.90
MLM			3.27	0.27	0.79	0.15	0.85
Prefix			3.24	0.26	0.63	0.27	0.92
Span	Yes		3.07	0.25	0.70	0.21	0.89
MLM			3.06	0.27	0.76	0.18	0.87
Prefix			3.11	0.26	0.78	0.19	0.89
Span	No	300	3.59	0.12	0.05	0.26	1.01
MLM			3.96	0.14	0.07	0.20	0.98
Prefix			3.28	0.11	0.05	0.38	1.00
Span	Yes		3.06	0.14	0.28	0.27	0.97
MLM			3.37	0.15	0.15	0.27	0.95
Prefix			3.11	0.13	0.22	0.33	0.99

Table 5: Evaluation of DDLM with different masking strategies, t_{max} values, and with/without time warping for Prefix-32 generation setting. The metrics with values < 0.5 (indicating highly repetitive samples) are displayed in colored font. We bolded the best metric values across other runs. See Section A for more details.

Enclosed-32							
Task	TW	t_{max}	AR-NLL	dist-1	MAUVE	self-BLEU	zipf
Data	-	-	3.29	N/A	N/A	0.09	0.86
Span	No	10	3.82	0.57	**0.92**	0.16	0.89
MLM			3.74	0.55	0.91	0.17	0.90
Prefix			3.89	0.57	0.91	0.16	0.88
Span	Yes		3.84	0.57	0.91	**0.15**	**0.87**
MLM			**3.69**	0.54	0.90	0.17	0.91
Prefix			3.86	**0.58**	0.91	0.16	0.90
Span	No	50	3.35	0.29	0.90	0.24	0.90
MLM			3.34	0.29	0.90	0.16	0.86
Prefix			3.41	0.27	0.90	0.30	0.94
Span	Yes		3.14	0.27	0.91	0.23	0.90
MLM			3.12	0.29	0.90	0.19	0.87
Prefix			3.26	0.27	0.90	0.21	0.89
Span	No	300	3.66	0.15	0.91	0.33	1.01
MLM			3.93	0.17	0.89	0.21	0.95
Prefix			3.40	0.12	0.90	0.40	1.02
Span	Yes		3.21	0.17	0.90	0.24	0.94
MLM			3.38	0.18	0.90	0.24	0.91
Prefix			3.25	0.14	0.90	0.34	1.00

Table 6: Evaluation of DDLM with different masking strategies, t_{max} values, and with/without time warping for Enclosed-32 generation setting. The metrics with values < 0.5 (indicating highly repetitive samples) are displayed in colored font. We bolded the best metric values across other runs. See Section A for more details.

L	H	D	Seq. len.	Masking	Optim.	Time Warping
8	8	1024	64	[MLM, Prefix, Span]	Adam	[no, yes]
LR	Scheduler	Warmup	Batch size	t_{max}	Steps	
3e-5	Cos. w/ Warmup	10k	1024	[10, 50, 300]	1e6	

Table 7: Pre-training hyperparameters used for experiments with noise scheduling (See Section A). L stands for number of layers, H for number of heads in the Transformer layer, and D for hidden size.