Stabilizing Diffusion LMs

Darshan Deshpande darshang@usc.edu

Shambhavi Sinha sinhasha@usc.edu

Manish Gawali gawali@usc.edu

Sreenidhi Iyengar Munimadugu munimadu@usc.edu

Maheshvar Chandrasekar mc64491@usc.edu

Abstract

Generative language models have made remarkable progress with the introduction of diffusion that leverage noising and denoising processes to model the data distribution. However, while convergence and stability issues have been addressed in the vision and audio domains, performance in the text domain remains unsatisfactory. To overcome these challenges, we propose leveraging recent work to incorporate anchor loss (Gao et al., 2022). In addition we use a custom noise scheduler which is a piece-wise linear noise function inspired by (Ho et al., 2020) into the diffusion model. This provides additional stability during training by constraining the denoising process to improve the performance of diffusion language models in the text domain. Our work facilitates high-quality and diverse text sample generation stably and efficiently. The code can be found here.

1 Introduction

The Diffusion Language Model (Diffusion LM) has recently emerged as a promising approach in the field of Natural Language Processing (NLP), garnering significant attention due to its potential in generating high-quality and coherent text (Li et al., 2022). This technique is based on the idea of iteratively refining the distribution of a given text based on its context and previous iterations using a diffusion process (Ho et al., 2020). This process involves a series of operations that iteratively smoothening the text distribution, ultimately leading to the generation of high-quality and coherent text.

In a Diffusion LM, the text is treated as a distribution of probability over a set of possible sequences of words (Li et al., 2022). The diffusion process is applied to this distribution, which starts from an initial distribution and iteratively updates the probability distribution of the text based on the context and previous iterations. This iterative process involves diffusing the distribution over a small number of timesteps, which allows the model to capture the long-range dependencies of the text (Ho et al., 2020).

Despite the success of Diffusion LMs in generating text, they still exhibit several limitations, such as slow convergence, unstable loss functions leading to embedding space collapsing and lack of customized noise schedules.

In order to address these limitations, we propose several approaches, including a piece-wise linear noise function (Ho et al., 2020) and a more stable anchor loss (Gao et al., 2022), aiming to improve the convergence and efficiency of Diffusion LMs for real-world applications (Li et al., 2022; Sohl-Dickstein et al., 2015).

2 Related work

In recent years, diffusion models have been shown to generate high-fidelity images and audio outputs by utilizing noising and denoising Markovian chains (Sohl-Dickstein et al., 2015). Notable examples of these capabilities include the works of Diffusion Beats GANs (Dhariwal and Nichol, 2021), DiffWave (Kong et al., 2020), and Riffusion (Forsgren and Martiros, 2022). Recent developments in this area include making the backward denoising chain non-Markovian (Song et al., 2020; Nichol and Dhariwal, 2021), and improving its diffusion to speed up generation while keeping high FID scores (Heusel et al., 2017).

The more recent transfer of diffusion models (Ho et al., 2020) from computer vision to text-to-vision domains (Ramesh et al., 2022; Rombach et al., 2021; Yu et al., 2022; Saharia et al., 2022; Oppenlaender, 2022) makes use of these advancements and presents a possibility for the application of diffusion in particular to text-based scenarios. Diffusion LMs were the first in this area to use diffusion to increase the controllability of language models. The DiffusionBERT design, which proposes a spindle schedule for noise addition and

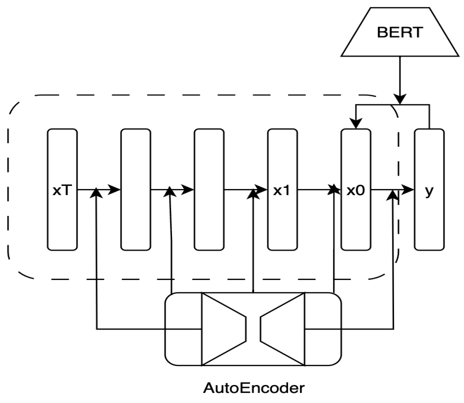

Figure 1: Diffusion LM

the concept of time agnostic decoding, and the Difformer architecture, which stabilizes the training by embedding norms and a novel anchor loss (Gao et al., 2022), respectively, further extended the framework of these LMs. SeqDiffuSeq (Yuan et al., 2022), a text diffusion model that combines an encoder-decoder Transformer architecture with self-conditioning and a novel adaptive noise schedule technique for sequence-to-sequence text generation, extends this idea to a sequence-to-sequence task and outperforms its encoder-only rival Diffuseq (Gong et al., 2022). (Strudel et al., 2022a) further tries to determine whether self conditioning is applicable to diffusion LMs. Latent Diffusion for Language Generation (Lovelace et al., 2022) presents a method to learn continuous diffusion models in the latent space of a pre-trained encoder-decoder language model, proposing an approach that views diffusion as a complementary method to existing pre-trained language models, enabling high-quality unconditional and class-conditional language generation. New pretraining techniques, including continuous paragraph denoising (Strudel et al., 2022b), are provided by more recent work on the scalability of such models and large-scale pretraining for greater downstream applicability of such models.

3 Problem Description

Diffusion LM (Li et al., 2022) has shown promise in generating high-quality and coherent text, particularly in non-autoregressive text generation. Despite its success, several limitations must be addressed to make it more practical for real-world applications. The main challenges we aim to tackle in this paper are:

1. **Slow convergence**: The convergence rate of Diffusion LM (Li et al., 2022) is relatively slow, which makes it less practical for real-world applications. The iterative process of diffusing the distribution over a small number of time steps captures long-range dependencies but at the cost of speed. Improving the convergence rate will make the model more efficient and suitable for various applications.

2. **Loss functions make the embedding space collapse**: Diffusion LM (Li et al., 2022) suffers from the collapse of the embedding space due to the usage of loss functions such as the rounding loss. This issue affects the overall performance and stability of the model and the quality of the generated text. By introducing a different loss function, the model can potentially achieve better stability and performance.

3. **Lack of customized noise schedules**: Diffusion LM (Li et al., 2022) does not have customized noise schedules, which limits its flexibility and adaptability to various text generation tasks. Incorporating a more flexible noise schedule will allow the model to adapt better to different tasks and improve its performance in diverse scenarios.

To address these limitations, we propose a set of approaches that aim to improve the convergence, efficiency, and flexibility of the Diffusion LM (Li et al., 2022). These approaches include:

- Introducing a piece-wise linear noise function (Ho et al., 2020) to replace the default square root noise schedule in Diffusion LM (Li et al., 2022). This custom noise schedule will allow the model to adapt better to different tasks and

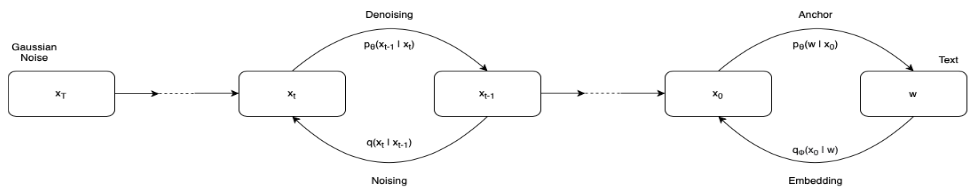

Figure 2: Diffusion LM

potentially improve its performance in diverse scenarios.

- Proposing a more stable anchor loss (Gao et al., 2022) that uses the clean model outputs instead of the noisy outputs used by the rounding loss. By using the clean outputs, the anchor loss can potentially achieve better stability and performance, leading to improved text generation quality.

4 Methods

The architecture combines two model pipelines: *D* and *B* where *D* consists of the diffusion pipeline and *B* is the BERT based encoder (Devlin et al., 2019). The diffusion pipeline *D* is structured similar to the implementation by (Ho et al., 2020) and the BERT encoder *B* is integrated into the main architecture similar to (Li et al., 2022) as shown in Figure 1.

4.1 End to End Diffusion

To perform diffusion, we first need to convert discrete textual data to a continuous space. We do this by projecting the text into a fixed dimensional embedding space by leveraging *B* which consists of a BERT encoder. We treat this embedding as the starting point for our diffusion pipeline *D* where we use a fixed dimensional autoencoder to denoise the embedding space at every diffusion reverse step. A diagramatic view of the process is shown in Figure 2

The noise is sampled using a piece-wise linear scheduler (Ho et al., 2020) which is defined as follows:

$$\begin{cases} linspace(\gamma * 0.0001 + 0.01, \gamma * 0.0001) \\ \quad \text{if } step \leq 10 \\ linspace(\gamma * 0.0001 + 0.01, \gamma * 0.02) \\ \quad \text{otherwise} \end{cases}$$

Here, γ is a hyperparameter which scales the noise in the schedule.

To convert the generated denoised embeddings from step x_0 to tokens y during the denoising process, we leverage a mapping function f_θ were θ are the learned weights. To constrain f_θ and stabilize training, (Li et al., 2022) proposed a rounding loss which is as follows:

$$L_{round} = -\log p_\theta(y|z_0)$$

But using this loss leads to embedding collapse in the early stages of training and hence we use the anchor loss (Gao et al., 2022) which is a slightly modified version of the rounding loss (Li et al., 2022) above. The anchor loss uses the prediction ($\bar{z_0}$) of the model instead of the noisy ground truth (z_0). The anchor loss can be formally written as:

$$L_{anchor} = -\log p_\theta(y|\bar{z_0})$$

5 Experimental Results

5.1 Datasets

We utilized two publicly available datasets from Huggingface namely Reuters (Apt'e et al., 1994) and 50% of CC News (Hamborg et al., 2017) due to computational limitations. Preprocessing of the textual data involved two main steps: cleaning and segmentation. The cleaning phase aimed at removing any noise or irrelevant information from the raw text data. After cleaning, the segmented text data was split into individual lines, a process known as linewise segmentation, to prepare it for tokenization.

5.2 Experimental Setup

We ran the model without initializing the weights from the BERT model (Devlin et al., 2019). The model was run using a learning rate of 0.0001 for 25,000 annealing steps and 2,000 diffusion steps with early stopping. A sequence length of 32 was used with a batch size of 64. The vocab size is 10,000 words. We used six different noise schedulers, including linear, cosine, square-root, truncated cosine, truncated linear, and piece-wise linear schedulers (Ho et al., 2020). Additionally, we used a weight decay of 0.0008 and a dropout rate of 0.1.

Dataset	Diffusion LM	Ours
Reuters Dataset	0.042 (Rounding)	**0.032 (Anchor)**
CC News Dataset (50%)	0.070 (Rounding)	**0.0617 (Anchor)**

Table 1: Experimental results comparing Diffusion LM and our proposed approach on the Reuters and CC News Datasets on MSE metric.

The experiments were conducted using three separate sets of resources: (1) 2x NVIDIA TESLA A40 GPUs (48 GB); (2) NVIDIA TESLA T4 GPU (16 GB); (3) NVIDIA Tesla A100 GPU (40 GB).

We are using the following repository for our implementation: (Madaan, 2022)

5.3 Results and Discussion

In this section, we present the results of our experiments and comapare it with the performance of (Li et al., 2022). We evaluate our models on two text generation tasks.

We firstly observe that using a anchor loss function is decreasing the mean-squared error value of the model when compared to using the rounding loss. This shows that our approach of 4ncluding the anchor loss is improving the quality of generation than (Li et al., 2022) for the Reuters dataset (Apt'e et al., 1994) and 50% of CC News dataset (Hamborg et al., 2017).

Figure 3 describes the rate of convergence of the model when initialized with different noise schedulers. We can see that using a piece-wise linear makes the model converge faster.

5.3.1 Model Performance

By introducing the piece-wise linear noise function (Ho et al., 2020), the stability and convergence rate of the Diffusion LM (Li et al., 2022) have been significantly improved. Our experiments show that the modified Diffusion LM converges faster than the original model, allowing us to generate high-quality text in less time.

The adoption of the more stable anchor loss (Gao et al., 2022) has effectively addressed the issue of embedding space collapse. Our findings indicate that the improved loss function results in a more stable training process, leading to better overall performance and higher-quality generated text.

5.3.2 Limitations and Challenges

While our proposed approaches have successfully addressed several limitations of the Diffusion LM (Li et al., 2022), there are still some challenges

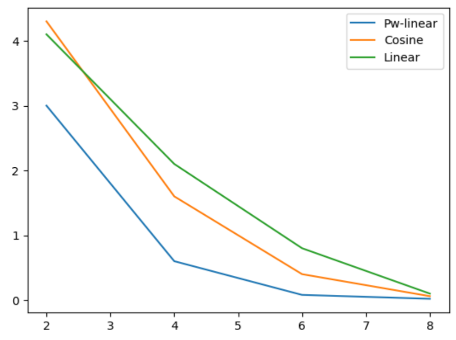

Figure 3: Convergence of different noise schedulers

that need to be considered. In our study, we observed that the decoding process of DDPMs (Ho et al., 2020) is computationally intensive and time-consuming. As a result, we aimed to investigate the potential benefits of utilizing DDIMs (Song et al., 2020) in terms of trade-offs between convergence rate and sample quality. However, due to time constraints, we were unable to conduct a comprehensive analysis of the efficacy of DDIMs for our specific use case. Additionally, the performance of our modified Diffusion LM (Li et al., 2022) on different text generation tasks and domains remains to be evaluated.

6 Conclusions and Future Work

In this study, we propose a set of approaches to enhance the performance of the Diffusion LM (Li et al., 2022) in text generation tasks. We observe better results by introducing a piece-wise linear noise scheduler (Ho et al., 2020) and adopting an anchor loss function (Gao et al., 2022).

Further research work can be done by incorporating DDIM (Song et al., 2020). This can potentially improve the execution time of models since it reduces the decoding time. However, it needs to be noted that the accuracy of such models can take a hit.

Future work could also explore the incorporation of additional techniques, such as adaptive noise schedules to further enhance the performance and controllability of the Diffusion LM (Li et al., 2022) in text generation and machine translation tasks.

7 Division of labor between the teammates

The work on this project was divided among the teammates as follows:

- **Maheshvar**: Conducted the literature review, designed and implemented the piece-wise linear noise function, performed hyperparameter tuning(Ho et al., 2020), and contributed to the writing of the problem description, experimental steup, and results sections.
- **Darshan**: Developed the more stable anchor loss (Gao et al., 2022), conducted experiments to evaluate the performance of the modified Diffusion LM (Li et al., 2022), and contributed to the writing of the results and discussion sections. Provided support in the analysis of experimental results.
- **Shambhavi**: Worked on reducing the memory space of the code and making it more efficient. Performed Cleaning and pre-processing of the datasets. Worked on multiple different noise schedulers to improve performance and contributed to the writing of the recent works, method, conclusion and future work sections.
- **Sreenidhi**: Assisted in the implementation of the piece-wise linear noise function (Ho et al., 2020) and the stable anchor loss (Gao et al., 2022) and contributed to the writing of the methodology, problem description and results sections.
- **Manish**: Made the code efficient with working on a reduced RAM space and conducted experiments to evaluate the performance of the modified Diffusion LM (Li et al., 2022) on different tasks and domains, and contributed to the writing of the discussion and conclusions sections.

All team members actively participated in discussions, provided feedback, and contributed to the project's overall direction. The final report was reviewed and edited collaboratively by all teammates.

References

Chidanand Apt'e, Fred Damerau, and Sholom M. Weiss. 1994. Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems*.

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.

Prafulla Dhariwal and Alex Nichol. 2021. Diffusion models beat gans on image synthesis.

Seth* Forsgren and Hayk* Martiros. 2022. Riffusion - Stable diffusion for real-time music generation.

Zhujin Gao, Junliang Guo, Xu Tan, Yongxin Zhu, Fang Zhang, Jiang Bian, and Linli Xu. 2022. Difformer: Empowering diffusion models on the embedding space for text generation.

Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and LingPeng Kong. 2022. Diffuseq: Sequence to sequence text generation with diffusion models. *arXiv preprint arXiv:2210.08933*.

Felix Hamborg, Norman Meuschke, Corinna Breitinger, and Bela Gipp. 2017. news-please: A generic news crawler and extractor. In *Proceedings of the 15th International Symposium of Information Science*, pages 218–223.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium.

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.

Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. 2020. Diffwave: A versatile diffusion model for audio synthesis.

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori Hashimoto. 2022. Diffusion-lm improves controllable text generation. *ArXiv*, abs/2205.14217.

Justin Lovelace, Varsha Kishore, Chao Wan, Eliot Shekhtman, and Kilian Weinberger. 2022. Latent diffusion for language generation. *arXiv preprint arXiv:2212.09462*.

Aman Madaan. 2022. Minimal text diffusion.

Alex Nichol and Prafulla Dhariwal. 2021. Improved denoising diffusion probabilistic models.

Jonas Oppenlaender. 2022. The creativity of text-to-image generation. In *25th International Academic Mindtrek conference*. ACM.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical text-conditional image generation with clip latents.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2021. High-resolution image synthesis with latent diffusion models.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. 2022. Photorealistic text-to-image diffusion models with deep language understanding.

Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep unsupervised learning using nonequilibrium thermodynamics.

Jiaming Song, Chenlin Meng, and Stefano Ermon. 2020. Denoising diffusion implicit models.

Robin Strudel, Corentin Tallec, Florent Altché, Yilun Du, Yaroslav Ganin, Arthur Mensch, Will Grathwohl, Nikolay Savinov, Sander Dieleman, Laurent Sifre, and Rémi Leblond. 2022a. Self-conditioned embedding diffusion for text generation.

Robin Strudel, Corentin Tallec, Florent Altché, Yilun Du, Yaroslav Ganin, Arthur Mensch, Will Grathwohl, Nikolay Savinov, Sander Dieleman, Laurent Sifre, and Rémi Leblond. 2022b. Self-conditioned embedding diffusion for text generation.

Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, Ben Hutchinson, Wei Han, Zarana Parekh, Xin Li, Han Zhang, Jason Baldridge, and Yonghui Wu. 2022. Scaling autoregressive models for content-rich text-to-image generation.

Hongyi Yuan, Zheng Yuan, Chuanqi Tan, Fei Huang, and Songfang Huang. 2022. Seqdiffuseq: Text diffusion with encoder-decoder transformers. *arXiv preprint arXiv:2212.10325*.