DiffusionBERT: Improving Generative Masked Language Models with Diffusion Models

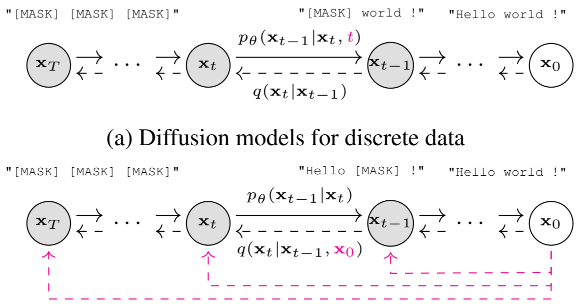
Zhengfu He*[*] Tianxiang Sun[*] Qiong Tang Kuanning Wang
Xuanjing Huang Xipeng Qiu[†]
School of Computer Science, Fudan University
Shanghai Key Laboratory of Intelligent Information Processing, Fudan University
{zfhe19,txsun19,wangkn20,xjhuang,xpqiu}@fudan.edu.cn
qtang22@m.fudan.edu.cn

Abstract

We present DiffusionBERT, a new generative masked language model based on discrete diffusion models. Diffusion models and many pre-trained language models have a shared training objective, i.e., *denoising*, making it possible to combine the two powerful models and enjoy the best of both worlds. On the one hand, diffusion models offer a promising training strategy that helps improve the generation quality. On the other hand, pre-trained denoising language models (e.g., BERT) can be used as a good initialization that accelerates convergence. We explore training BERT to learn the reverse process of a discrete diffusion process with an absorbing state and elucidate several designs to improve it. First, we propose a new noise schedule for the forward diffusion process that controls the degree of noise added at each step based on the information of each token. Second, we investigate several designs of incorporating the time step into BERT. Experiments on unconditional text generation demonstrate that DiffusionBERT achieves significant improvement over existing diffusion models for text (e.g., D3PM and Diffusion-LM) and previous generative masked language models in terms of perplexity and BLEU score. Promising results in conditional generation tasks show that DiffusionBERT can generate texts of comparable quality and more diverse than a series of established baselines.

1 Introduction

Diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021) have recently emerged as a new class of state-of-the-art generative models, achieving high-quality synthesis results on image data (Ramesh et al., 2022; Rombach et al., 2022; Saharia et al., 2022). Though these models captured widespread attention from not only the research community but also the public, applying diffusion models to text data is still challenging and under-explored due to the discrete nature of the text. A few prior works that explored using diffusion models on text data can be divided into two lines. The first is to extend diffusion models to discrete state spaces (Hoogeboom et al., 2021; Austin et al., 2021). The second is to perform the diffusion process and its reverse process in the continuous domain and bridge the continuous and the discrete domain through embedding and rounding (Li et al., 2022; Gong et al., 2022). However, none of these works leveraged pre-trained language models (PLMs, Devlin et al. (2019); Lewis et al. (2020); Raffel et al. (2020); Brown et al. (2020); Qiu et al. (2020)), which are an unmissable treasure in the NLP community.


"[MASK] [MASK] [MASK]"
"[MASK] world !"
"Hello world !"
$\mathbf{x}_T$
$\mathbf{x}_t$
$\mathbf{x}_{t-1}$
$\mathbf{x}_0$
$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, t)$
$q(\mathbf{x}_t|\mathbf{x}_{t-1})$
"[MASK] [MASK] [MASK]"
"Hello [MASK] !"
"Hello world !"
$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$
$q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)$

(a) Diffusion models for discrete data

(b) Non-Markovian DiffusionBERT

Figure 1: In contrast to conventional discrete diffusion models, DiffusionBERT uses BERT as its backbone to perform text generation. The main differences are highlighted in color: (1) DiffusionBERT performs decoding without knowing the current time step while canonical diffusion models are conditioned on time step. (2) The diffusion process of DiffusionBERT is non-Markovian in that it generates noise samples $\mathbf{x}_t$ conditioning not only on $\mathbf{x}_{t-1}$ but also on $\mathbf{x}_0$. Such a non-Markov process is due to our proposed noise schedule.

This work, to our knowledge, is the first attempt to combine diffusion models with PLMs. Such a combination is built upon a shared training ob-

[*] Equal contribution.
[†] Corresponding author.

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 4521–4534
July 9-14, 2023 ©2023 Association for Computational Linguistics

jective between diffusion models and PLMs, i.e., *denoising*. Diffusion models consist of a forward process (data to noise) and a reverse process (noise to data). In the forward process, a small amount of noise is gradually added to the data. Then, a neural network (p_θ in Figure 1) is employed to learn the reverse process step by step, i.e., learn to denoise. Such a denoising neural network is naturally related to a wide class of PLMs that are pre-trained with denoising objectives such as BERT (Devlin et al., 2019) and BART (Lewis et al., 2020). Hence, pre-trained denoising language models can serve as a good starting point to learn the reverse diffusion process. On the other hand, diffusion models also offer a promising training strategy for generative PLMs. In contrast to commonly used generative PLMs (e.g., GPT (Brown et al., 2020)) that rely on an autoregressive factorization of the joint probability, diffusion models provide another way of factorization along the dimension of time and therefore allow the generative model to be not necessarily autoregressive. Thus, diffusion models can be combined with a variety of PLMs that may not be pre-trained for generation.

In the discrete domain, the forward diffusion process can be implemented by a chain of transition matrices that gradually corrupt the clean text. As shown in Figure 1, the clean text "`Hello world !`" is gradually corrupted into "`[MASK] [MASK] [MASK]`" during the diffusion process. In this work, we explore using pre-trained denoising language models (e.g., BERT) to learn the reverse diffusion process and demonstrate their advantages in accelerating convergence and improving generation quality. Further, we propose a new noise schedule of the forward process based on the principle of distributing the corrupted information uniformly across the forward process. The noise schedule, called *spindle schedule*, generates noise for $\mathbf{x}_t$ conditioned not only on $\mathbf{x}_{t-1}$ but also on $\mathbf{x}_0$, making the forward process non-Markovian without changing the original training objective. Note that the denoising model takes as input $\mathbf{x}_t$ and time step t to predict $\mathbf{x}_{t-1}$, where t is unseen during the pre-training of language models so we investigate several ways of incorporating the time step into PLMs. As a result, we find that the best result is achieved by throwing away the time information, which we call *time-agnostic decoding* (TAD).

Experimental results on unconditional text generation demonstrate the benefit of combining diffusion models with PLMs: the proposed DiffusionBERT significantly improves the generation quality over existing diffusion models for text generation (e.g., D3PM (Austin et al., 2021) and Diffusion-LM (Li et al., 2022)) and previous generative masked language models (e.g., BERT-Mouth (Wang and Cho, 2019)). DiffusionBERT also matches several strong baselines in conditional generation tasks and shows superior generation diversity. The effectiveness of the proposed spindle schedule and time-agnostic decoding is confirmed by ablation studies. In a nutshell, DiffusionBERT enjoys the best of both worlds.

2 Background

2.1 Diffusion Models

Diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020) are a class of latent variable models that are originally designed for continuous domains. A diffusion model is consisting of a forward diffusion process and a reverse diffusion process. Given a sample $\mathbf{x}_0 \sim q(\mathbf{x}_0)$, a Markov chain of latent variables $\mathbf{x}_1, \cdots, \mathbf{x}_T$ are produced in the forward process by progressively adding a small amount of Gaussian noise to the sample:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}), \quad (1)$$

where $\{\beta_t \in (0,1)\}_{t=1}^T$ is a noise schedule controlling the step size of adding noise. Eventually $\mathbf{x}_T$ becomes an isotropic Gaussian distribution. If β_t is small enough, the reverse process $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ is also a Gaussian, which is learned by a parameterized model

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t)), \quad (2)$$

where $\mu_\theta(\cdot)$ and $\Sigma_\theta(\cdot)$ can be implemented by a U-Net or a Transformer. When conditioning also on $\mathbf{x}_0$, $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$ has a closed form so we can manage to minimize the variational lower bound to optimize $\log p_\theta(\mathbf{x}_0)$:

$$\begin{aligned}\mathcal{L}_{\text{vlb}} &= \mathbb{E}_q[D_{\text{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p_\theta(\mathbf{x}_T))] \\ &+ \mathbb{E}_q[\sum_{t=2}^{T} D_{\text{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, t))] \\ &- \log p_\theta(\mathbf{x}_0|\mathbf{x}_1), \quad (3)\end{aligned}$$

where $\mathbb{E}_q(\cdot)$ denotes the expectation over the joint distribution $q(\mathbf{x}_{0:T})$.

2.2 Diffusion Models in Discrete Domain

For discrete domains, each element of $\mathbf{x}_t$ is a discrete random variables with K categories. For text data, $K = |V|$ is the size of the vocabulary. Denote $\mathbf{x}_t$ as a stack of one-hot vectors, the process of adding noise can be written as

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathtt{Cat}(\mathbf{x}_t; \mathbf{p} = \mathbf{x}_{t-1}\mathbf{Q}_t), \quad (4)$$

where $\mathtt{Cat}(\cdot)$ is a category distribution and $\mathbf{Q}_t$ is a transition matrix that is applied to each token in the sequence independently: $[\mathbf{Q}_t]_{i,j} = q(x_t = j|x_{t-1} = i)$. It is easy to obtain that

$$\begin{aligned} q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) &= \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)} \\ &= \mathtt{Cat}\left(\mathbf{x}_{t-1}; \mathbf{p} = \frac{\mathbf{x}_t\mathbf{Q}_t^\top \odot \mathbf{x}_0\overline{\mathbf{Q}}_{t-1}}{\mathbf{x}_0\overline{\mathbf{Q}}_t\mathbf{x}_t^\top}\right), \quad (5) \end{aligned}$$

where $\overline{\mathbf{Q}}_t = \mathbf{Q}_1\mathbf{Q}_2 \cdots \mathbf{Q}_t$. Note that $\odot$ is element-wise multiplication and the division is row-wise.

With $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$ at hand, according to Eq. (3), we can use a parameterized model $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, t)$ to learn the reverse diffusion process.

3 DiffusionBERT

In contrast to recently proposed diffusion models for text, e.g., Diffusion-LM (Li et al., 2022) and DiffuSeq (Gong et al., 2022), which are based on *continuous* diffusion models, we instead explore *discrete* diffusion models to integrate PLMs as the backbone. We first introduce a specific instance of discrete diffusion models (Austin et al., 2021), which considers a transition matrix with an absorbing state for the sake of using PLMs (§ 3.1). Secondly, we introduce a new noise schedule of the forward diffusion process, called spindle schedule, which is based on the principle of distributing the corrupted information uniformly across the forward process (§ 3.2). Then, we investigate several alternatives of incorporating the time step into PLMs for predicting $\mathbf{x}_{t-1}$ given $\mathbf{x}_t$ and t (§ 3.3). Finally, we explore training DiffusionBERT for conditional generation with prompts (§ 3.4).

3.1 Diffusion Models with a Discrete Absorbing State

To be combined with pre-trained denoising language models, we incorporate an *absorbing state*, e.g., `[MASK]` for BERT, in the Markov process. In particular, each token in the sequence either stays the same or transitions to `[MASK]` with some probability. Formally, each entry of the transition matrix at step t is as follows,

$$[\mathbf{Q}_t]_{i,j} = \begin{cases} 1 & \text{if } i = j = \mathtt{[M]}, \\ \beta_t & \text{if } j = \mathtt{[M]}, i \neq \mathtt{[M]}, \\ 1 - \beta_t & \text{if } i = j \neq \mathtt{[M]}, \end{cases} \quad (6)$$

where `[M]` is the abbreviation of `[MASK]`. Such a Markov process converges to a stationary distribution $q(\mathbf{x}_T)$, which places all probability mass on a sequence with all `[MASK]` tokens.

The t-step marginal $q(\mathbf{x}_t^i|\mathbf{x}_0^i)$ can be easily obtained in a closed form,

$$q(\mathbf{x}_t^i|\mathbf{x}_0^i) = \begin{cases} \overline{\alpha}_t & \text{if } \mathbf{x}_t^i = \mathbf{x}_0^i, \\ 1 - \overline{\alpha}_t & \text{if } \mathbf{x}_t^i = \mathtt{[M]}, \end{cases} \quad (7)$$

where $\overline{\alpha}_t = \prod_{i=1}^t (1 - \beta_i)$, $\mathbf{x}_t^i$ denotes the i-th token in the sequence at step t. Combining with Eq. (3) and (5), we can derive a training objective to optimize $p_\theta(\mathbf{x_{t-1}}|\mathbf{x_t}, t)$ and generate a sample by performing the reverse diffusion process:

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T)\prod_{t=1}^T p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, t). \quad (8)$$

3.2 Spindle Noise Schedule

The noise schedule in the continuous domain, such as the linear schedule (Ho et al., 2020) and the cosine schedule (Nichol and Dhariwal, 2021), has shown to be important to the performance of diffusion models.

In contrast to the continuous domain where the noise can be easily controlled by the variance of the Gaussian, *(1) it is less obvious how to control the degree of noise added at each step in the discrete domain*. For the discrete domain, the noise schedule $\beta_t = (T - t + 1)^{-1}$ has been explored for the case of the uniform transition matrix (Sohl-Dickstein et al., 2015; Hoogeboom et al., 2021) and the absorbing-state transition matrix (Austin et al., 2021). However, *(2) such a schedule assumes all tokens carry the same amount of information and does not consider the linguistic difference among the tokens in a sequence*. Besides, *(3) it violates the easy-first-generation nature of denoising language models*. That is, the model tends to generate tokens that are most frequently appearing (and is least surprising) in the training corpus to achieve a higher likelihood. As the context becomes richer, more details come up in the sequence.

To address the above issues, we consider a noise schedule that (1) measures the added noise at each step by the corrupted information and encourage the corrupted information to be uniformly distributed across the diffusion steps. Since the information is measured independently for each token, (2) different tokens in a sequence are assigned different probabilities of transitioning to the `[MASK]` token. Moreover, inspired by the easy-first-generation phenomenon, (3) we put the tokens in a sequence in descending order of their information and divide them into T buckets. Each bucket is ensured to contain the same amount of information. That is, we mask the most informative tokens at the start of the forward process and mask the least informative tokens at the end of the forward process such that the learnable reverse process follows an easy-first generative behavior.

In particular, distributing corrupted information uniformly across the forward steps can be formally described by

$$1 - \frac{t}{T} = \frac{\sum_{i=1}^{n} H(\mathbf{x}_t^i)}{\sum_{i=1}^{n} H(\mathbf{x}_0^i)} = \frac{\sum_{i=1}^{n} \overline{\alpha}_t^i H(\mathbf{x}_0^i)}{\sum_{i=1}^{n} H(\mathbf{x}_0^i)}, \quad (9)$$

where H denotes the entropy, which measures the amount of information of a random variable, $\mathbf{x}^i$ denotes the i-th token in the sequence and n denotes the length of the sequence. According to Eq. (7), $\overline{\alpha}_t^i = \prod_{j=1}^{t}(1-\beta_j^i)$ denotes the probability that the i-th token remains the same at step t, i.e., $\mathbf{x}_t^i = \mathbf{x}_0^i$. We expect that $\overline{\alpha}_t^i > \overline{\alpha}_t^j$ if $H(\mathbf{x}_t^i) < H(\mathbf{x}_t^j)$ such that easy (low-information) tokens emerges earlier than hard (high-information) tokens during the reverse process. In practice, the entropy of a given token $H(\mathbf{x})$ is calculated by the negative logarithm of its frequency in the training corpus.

Considering these aforementioned properties, we construct $\overline{\alpha}_t^i$ as follows,

$$\overline{\alpha}_t^i = 1 - \frac{t}{T} - S(t) \cdot \tilde{H}(\mathbf{x}_0^i), \quad (10)$$

$$S(t) = \lambda \sin \frac{t\pi}{T}, \quad (11)$$

$$\tilde{H}(\mathbf{x}_0^i) = 1 - \frac{\sum_{j=1}^{n} H(\mathbf{x}_0^j)}{nH(\mathbf{x}_0^i)}, \quad (12)$$

where $S(t)$ is introduced to control the effect of the informativeness at time step t. It is designed to be sinusoidal to ensure $S(0) = S(T) = 0$ such that $\mathbf{x}_t$ can retain all (zero) information when $t = 0$ ($t = T$). The effect of $S(t)$ is controlled by a hyperparameter λ. When $\lambda = 0$, the noise schedule is degraded to $\beta_t = (T-t+1)^{-1}$ as in Sohl-Dickstein et al. (2015); Hoogeboom et al. (2021); Austin et al. (2021). Figure 2 shows how $\overline{\alpha}$ progresses during the forward process. The schedule is named as *spindle* due to the shape of the probability curves.

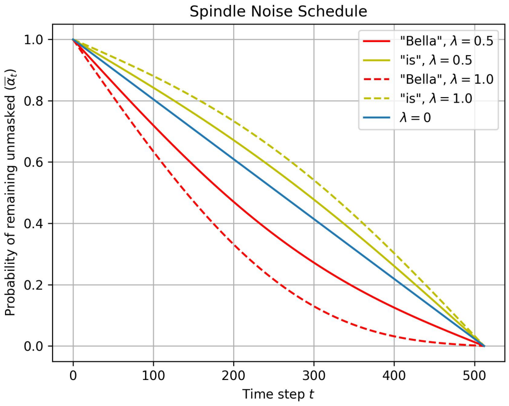


Figure 2: Each token in a sequence has a specific noise schedule depending on how much information is lost when they are masked. For instance, in the sentence `"Bella is sitting over there."`, `"Bella"` is the most informative word. Thus it is encouraged to be masked at the early stage so that our model learns to recover it in the last place.

In our proposed schedule, the transition probability at time step t depends not only on the current state but also on the original text, making the forward diffusion process non-Markovian. Nevertheless, as revealed by Eq. (5), this does not change the original training objective.

3.3 The Design Space of Feeding Time Steps

Typically, a diffusion model takes as input a noised sample and the time step to predict the denoised sample during the reverse process, i.e., $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, t)$. However, t is an additional variable that is unseen during the pre-training of language models and therefore it is less trivial how to feed the time information into the PLMs. Here we explore three design choices of feeding time steps.

Layer-wise Time Embedding A straightforward choice is to include the time step as the same way as positional encoding, i.e., using the Transformer sinusoidal embedding or a learnable MLP in each Transformer layer. Note that this way is commonly adopted in previous work (Ho et al., 2020; Austin et al., 2021; Li et al., 2022).

Prefix Time Embedding Prompting language models by prepending trainable soft tokens to the input sequence has shown promising results recently (Lester et al., 2021; Sun et al., 2022). Hence, we also explore including a time step token embedding $\mathbf{v}(t)$ as a prefix of the input token embeddings $\langle \mathbf{v}(\mathbf{x}_t^1), \mathbf{v}(\mathbf{x}_t^2), \cdots, \mathbf{v}(\mathbf{x}_t^n) \rangle$. In particular, the time step token is inserted in between the `[CLS]` token and the input sequence. These added time step token embeddings are trained along with the PLM.

Time-Agnostic Decoding Another alternative is not to explicitly incorporate the time step t because it can be implied by the noised sample $\mathbf{x}_t$. In contrast to the image data, it is easier to implicitly infer the diffusion time step by counting the number of corrupted tokens (i.e., `[MASK]`) in the noised sequence. In this way, the PLM has to perform iterative decoding while being ignorant of the current time step, i.e., $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$.

3.4 Prompting DiffusionBERT for Conditional Generation

An intriguing property of PLMs is understanding instructions provided in the context and performing desired tasks according to the instructions. Inherited from BERT, DiffusionBERT is also able to solve a wide range of conditional generation tasks by prompting with task descriptions and task texts to be processed (see Appendix.B for examples). To explore the DiffusionBERT for conditional generation, we adopt partial denoising (Gong et al., 2022) to perform the diffusion process conditioning on the partially corrupted text.

4 Experiments

4.1 Tasks and Datasets

We train DiffusionBERT on the One Billion Word dataset (LM1B) (Chelba et al., 2014) for unconditional generation. LM1B is a corpus with about 30 million sentences and a vocabulary of about 793k. For conditional generation, we choose two tasks for evaluating DiffusionBERT, namely Question Generation (QG) and Paraphrasing. Quasar-T (Dhingra et al., 2017) is a Question Answering dataset containing enormous document-question pairs. Gong et al. (2022) constructed a QG dataset from Quasar-T and we follow their data split and task settings. For paraphrasing, we choose Quora Question Pairs (QQP)[1], a widely used question-pairs dataset with 147K training samples.

4.2 Baselines

We conduct comparison on unconditional text generation against several non-autoregressive (NAR) baselines: D3PM (Austin et al., 2021), Diffusion-LM (Li et al., 2022), and BERT-Mouth (Wang and Cho, 2019). We consider DiffuSeq (Gong et al., 2022) as a baseline for conditional generation. Several strong baselines reported in their paper are also included for further comparison.

D3PM D3PM is a general framework of discrete diffusion models. We implement an instance of D3PM with the absorbing state and a layer-wise time embedding. Both DiffusionBERT and D3PM are implemented with a sequence length $n = 64$ and diffusion steps $T = 2048$. During inference, we perform DDIM sampling with time step size of 16 in each iteration. Hence, the total inference cost is 128 iterations.

Diffusion-LM Diffusion-LM learns an embedding to map discrete text into the continuous space where it performs Gaussian diffusion process. A rounding step is required to map the continuous embeddings into discrete texts. We re-implemented Diffusion-LM with the model architecture of BERT and diffusion steps $T = 2000$. Since the performance drop of Diffusion-LM is bigger than DiffusionBERT when sampling less steps, we do not skip steps during generation.

BERT-Mouth BERT-Mouth samples text from BERT via order-agnostic autoregressive masked language modeling. Starting from a sequence of `[MASK]`, BERT samples one token at each time step in random order. We continue training BERT on LM1B for fair comparison.

DiffuSeq DiffuSeq introduces a conditional text generation framework for encoder-only diffusion models. It performs diffusion process only on the target (i.e. partial denoising). Experimental results of DiffuSeq show that generation of diffusion models is generally more diverse in conditional generation settings. Such diversity contributes to sample quality with the help of Minimum Bayes Risk (MBR) decoding (Koehn, 2004) i.e., sampling multiple candidates and re-ranking them by their BLEU scores relative to other candidates.

[1] https://www.kaggle.com/c/quora-question-pairs

Method	Pretrained	Schedule	Time Step	PPL ↓	BLEU ↑	Self-BLEU ↓
D3PM (Austin et al., 2021)	✗	$(T-t+1)^{-1}$	LTE	82.34	0.3897	0.2347
			TAD	125.15	0.3390	0.2720
		Spindle	LTE	77.50	0.4241	0.2288
Diffusion-LM (Li et al., 2022)	✗	Cosine	LTE	118.62	0.3553	0.2668
	✓	Cosine	LTE	132.12	0.3562	0.2798
BERT-Mouth (Wang and Cho, 2019)	✓	-	-	142.89	0.2867	**0.1240**
DiffusionBERT	✓	$(T-t+1)^{-1}$	LTE	92.53	0.3995	0.2118
			PTE	79.95	0.3886	0.2156
			TAD	78.76	0.4213	0.2116
		Spindle	TAD	**63.78**	**0.4358**	0.2151

Table 1: Main results on LM1B. The methods proposed in this work are marked with wavy lines. The best results are in **bold** and the second best results are underlined. LTE: layer-wise time embedding. PTE: prefix time embedding. TAD: time-agnostic decoding.

4.3 Experimental Setup

In both conditioned and unconditioned settings, our DiffusionBERT is based on BERT-BASE-UNCASED with about 110M parameters. We train DiffusionBERT using the AdamW optimizer (Loshchilov and Hutter, 2019) with learning rate of 3e-6, dropout probability of 0.1 and batch size of 256. We use a 10K-step linear warmup schedule starting from learning rate of 1e-8. For generation efficiency and better alignment with BERT pre-training objective, we use DDIM sampling (Song et al., 2021) in which BERT generates all tokens at first and performs the forward process in Eq. 7 to skip time steps. All experiments are conducted on NVIDIA A100 Tensor Core GPUs. We use 4 GPUs for training and a single GPU for sampling.

To sample from DiffusionBERT trained on LM1B, we use a top-K filter with $K = 30$ and perform 128 steps of inference to align with the settings in Austin et al. (2021).

In the two conditional generation tasks, DiffusionBERT is trained for 100K steps. Sampling involves a top-15 filter and 400 inference steps. We follow Gong et al. (2022) in the MBR size $|S| = 10$ and sequence length of 128.

4.4 Unconditional Generation

Our main results of unconditional sampling are included in Table 1. We choose BLEU-4 and self-BLEU-4 (Zhu et al., 2018) as the metric for generation quality and diversity, respectively. In particular, we follow Savinov et al. (2022); Caccia et al. (2020) to sample 1K sentences to compute BLEU score relative to all sentences in the test set. Another 1K sentences are sampled for computing self-BLEU. Overall, DiffusionBERT achieves the best generation quality and diversity trade-off among the considered NAR methods. Besides, the perplexity of DiffusionBERT with the spindle noise schedule is substantially lower. Evidence of lower bound is used as a proxy of the perplexity of DiffusionBERT and D3PM since the exact likelihood of diffusion models is intractable.

DiffusionBERT vs. Other Generative BERT Models We compare DiffusionBERT with another representative generative masked language model, BERT-Mouth (Wang and Cho, 2019). Experimental results show that DiffusionBERT achieves better performance in terms of the perplexity and the BLEU score. We attribute the superior performance of DiffusionBERT to its one-time sampling of all tokens, which helps DiffusionBERT generate more coherent text, especially in a long range. Although such decoding may face the problem of multimodality (Gu et al., 2018), inappropriate phrases can be fixed in the upcoming diffusion steps. The probabilistic modeling offers more flexibility in that generated tokens with low probability are more likely to be masked and resampled. Wang and Cho (2019) also proposed to continue masking and predicting tokens after the whole sequence is complete. But such randomness in the selection and replacement of tokens results in low inference speed.

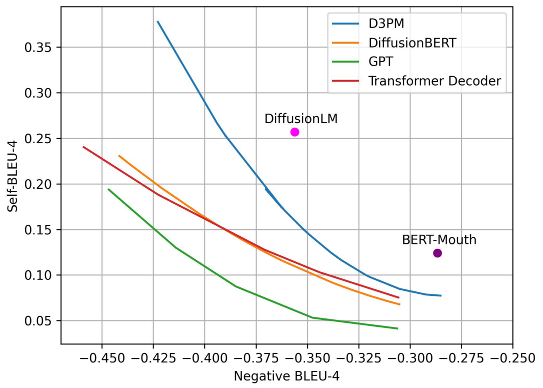


Figure 3: BLEU scores on the LM1B test set. Left is better, lower is better.

Discrete vs. Continuous Diffusion Models We then focus on the comparison of discrete and continuous diffusion models for text generation. To achieve this, we mainly compare DiffusionBERT with Diffusion-LM, which is based on continuous diffusion models. As a result, despite of its outstanding controlling ability, we show that the texts generated by Diffusion-LM have a lower quality than DiffusionBERT. Though both DiffusionBERT and Diffusion-LM adopt the same configuration of Transformer, it is worth noting that the superior performance of DiffusionBERT may be contributed by not only the discrete diffusion models but also the use of pre-trained models. To disentangle the effect of pre-training and discrete/continuous diffusion models, we also explore initializing Diffusion-LM with BERT. As shown in Table 1, training Diffusion-LM from BERT initialization performs even worse than training from scratch. We conjecture that the continuous nature of Diffusion-LM is not compatible with the initialization from BERT since the embedding learned by BERT may not be suitable for the Gaussian diffusion process. In contrast, the comparison of D3PM and DiffusionBERT shows that DiffusionBERT benefits much from the BERT initialization due to its discrete diffusion process.

Effect of Time Step In terms of both likelihood and generation quality, the layer-wise time embedding (LTE) lags far behind the other two time step designs for DiffusionBERT while time-agnostic decoding (TAD) achieves the best result. By contrast, D3PM without time step embedding performs significantly worse. In a nutshell, simplifying time step design has positive effect on DiffusionBERT but is quite harmful for D3PM. This suggests that initializing p_θ with PLMs enables DiffusionBERT to perform generation without explicitly providing time information yet achieving better generation results. The resemblance between BERT pre-training objective and absorbing diffusion models makes it easier for DiffusionBERT to generalize to noisier scenarios while a Transformer encoder trained from scratch needs a specific time-aware module to model the reverse process.

Method/Metric	Quality		Diversity	
	BLEU ↑	**Rouge-L ↑**	**Self-BLEU ↓**	**Div-4 ↑**
GRU-attention	0.0651	0.2617	0.9999	0.3178
Transformer-base	0.0364	0.1994	0.8767	0.4055
GPT2-base FT	0.0741	0.2714	0.1403	0.9216
GPT2-large FT	0.1110	0.3215	0.2910	0.8062
GPVAE-T5	0.1251	0.3390	0.3567	0.7282
NAR-LeVT	0.093	0.2893	0.983	0.4776
DiffuSeq	**0.1731**	**0.3665**	0.2789	0.8103
DiffusionBERT	0.0971	0.3420	**0.0703**	**0.9372**

(a) QG

Method/Metric	Quality		Diversity	
	BLEU ↑	**Rouge-L ↑**	**Self-BLEU ↓**	**Div-4 ↑**
GRU-attention	0.1894	0.5129	0.9958	0.3287
Transformer-base	0.0580	0.2489	0.7717	0.4312
GPT2-base FT	0.1980	0.5212	0.5480	0.6245
GPT2-large FT	0.2059	0.5415	0.7325	0.5020
GPVAE-T5	0.2409	**0.5886**	0.5604	0.6169
NAR-LeVT	0.2268	0.5795	0.9995	0.3329
DiffuSeq	0.2413	0.5880	0.2732	0.8641
DiffusionBERT	**0.2420**	0.5845	**0.1504**	**0.9770**

(b) Paraphrase

Table 2: Results on conditional generation tasks. The best and second best results are remarked in **bold** and underlined, respectively.

Effect of the Spindle Noise Schedule We try our proposed spindle noise schedule on both DiffusionBERT and D3PM. The perplexity is improved by 5.8% and 19% for D3PM and DiffusionBERT, respectively. Besides, D3PM with the spindle schedule outperforms that with the standard $(T-t+1)^{-1}$ schedule in terms of BLEU score. The same trend holds for DiffusionBERT but with a smaller margin.

4.5 Quality-Diversity Trade-off

Figure 3 demonstrates the quality-variation tradeoff by changing the truncation parameter K or the sampling temperature τ in 10 values[2], DiffusionBERT exhibits comparable generation ability with

[2] Except for Diffusion-LM and BERT-Mouth since controlling temperature has little effect.

a Transformer decoder trained from scratch and pushes the Pareto front of NAR generation quality/diversity trade-off by a large margin. However, it still falls behind pre-trained AR models of the same size.

4.6 Training Efficiency

One important feature of DiffusionBERT is that with time-agnostic decoding, all parameters are initialized by pre-trained models. Consequently, the model includes fewer parameters and gets rid of adapting new parameters, improving the training efficiency. We only train DiffusionBERT for 40% steps of D3PM and 20% steps of DiffuSeq till convergence. Appendix.D provides a detailed comparison of convergence speed between DiffusionBERT and the diffusion baselines.

4.7 Conditional Generation

To evaluate the generation quality and diversity, we choose two metrics in each aspect. Besides BLEU and Self-BLEU scores, we also report Rouge-L for quality and Div-4 score for diversity. Higher Rouge-L (resp. Div-4) suggests better generation quality (resp. diversity).

As shown in Table 2, DiffusionBERT achieves competitive performance on both tasks and surpasses other baselines by a large margin in terms of diversity. We attribute such variety to the knowledge BERT obtained during pre-training and the diffusion generative process (Li et al., 2022). Apart from lexical or grammatical diversity, DiffusionBERT covers a wider range of semantic meanings. Moreover, such diversity contributes to generation quality via MBR decoding (Gong et al., 2022).

5 Related Work

5.1 BERT for Text Generation

It has been shown by Wang and Cho (2019) that the transfer-learning ability of BERT does not only helps to achieve impressive results in natural language understanding but also benefits sequential sampling for text generation. However, its bi-directionality nature holds BERT from matching the decoder-only counterparts (Radford et al., 2018) in modeling text from left to right.

5.2 Diffusion Models for Text

This work lies in the line of diffusion models, a latent variable generative framework proposed by Sohl-Dickstein et al. (2015). It has been architecturally improved by Ho et al. (2020) and has gained broad attention for its impressive generation ability and controllability in image generation (Ramesh et al., 2022; Saharia et al., 2022). Despite that, diffusion models for text still struggle to match autoregressive models in various generation tasks. Since the Gaussian noise proposed in Sohl-Dickstein et al. (2015) cannot be directly applied to discrete data, they also introduced a discrete forward process with a Bernoulli transition kernel. Hoogeboom et al. (2021) generalized from Bernoulli to categorical distributions. A more general family of discrete diffusion processes was introduced in Austin et al. (2021); Hoogeboom et al. (2022), including absorbing kernels and combinations of absorbing and uniform transition kernels. Li et al. (2022); Gong et al. (2022) models text in the continuous embedding space, which is closer to the settings in earlier works of diffusion models.

5.3 Non-Autoregressive Text Generation

Absorbing discrete diffusion models resembles conditional masked language models (Ghazvininejad et al., 2019) in that both methods predict the whole sequence simultaneously and follows a construct-destruct pattern to iteratively refine the generated text. The difference between those two models has been discussed in Austin et al. (2021). Savinov et al. (2022) proposed to approach the problem of non-autoregressive text modeling via unrolling the generation path, which resembles the idea of diffusion models for unconditional text generation. Non-autoregressive models are also considered in translation but implemented in various ways, e.g., insertion/deletion (Gu et al., 2019) and iterative sequence alignment (Saharia et al., 2020).

6 Conclusion

This work aims to approach the problem of text generation with non-autoregressive models. To achieve this, we combine pre-trained denoising language models with absorbing-state discrete diffusion models. The training procedure of our method includes two main deviations from current discrete diffusion models, i.e., a new family of time step designs and the spindle noise schedule. The spindle noise assigns a schedule for each token according to its frequency in the training corpus. Experimental results of unconditional generation demonstrate the success of DiffusionBERT in terms of generation quality and diversity. In constrained settings, DiffusionBERT surpasses 7 strong baselines in generation variety by a large margin and its generation quality matches state-of-the-art methods.

Limitations

In this work, we demonstrate the effectiveness of the proposed DiffusionBERT. However, the sampling efficiency in unconditional generation still lags behind fine-tuned GPT and we observe a few sampled sentences lacking coherence when the pre-assigned length is large (e.g., 128). The issue of inference efficiency is more severe in constrained settings in that MBR decoding samples multiple sentences for one source text. Though it brings significant improvement in BLEU and Rouge-L scores, the sampling time of one batch is several times that of unconditional generation.

Ethics Statement

The proposed DiffusionBERT is a novel approach for text-based diffusion models. In addition, we demonstrate that DiffusionBERT can achieve highly efficient training due to the use of PLMs. Therefore, this work helps reduce computation costs and carbon emissions. Though all the datasets and PLMs used in our experiments are publicly available and have not been reported to carry social bias against any sensitive attributes, more work is still needed to investigate the potential unfairness in these datasets and the knowledge BERT carries.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (No. 62236004 and No. 62022027).

References

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. 2021. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 17981–17993.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Massimo Caccia, Lucas Caccia, William Fedus, Hugo Larochelle, Joelle Pineau, and Laurent Charlin. 2020. Language gans falling short. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Ciprian Chelba, Tomás Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2014. One billion word benchmark for measuring progress in statistical language modeling. In *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore, September 14-18, 2014*, pages 2635–2639. ISCA.

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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Bhuwan Dhingra, Kathryn Mazaitis, and William W. Cohen. 2017. Quasar: Datasets for question answering by search and reading. *CoRR*, abs/1707.03904.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 6111–6120. Association for Computational Linguistics.

Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and Lingpeng Kong. 2022. Diffuseq: Sequence to sequence text generation with diffusion models. *CoRR*, abs/2210.08933.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems*

2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada, pages 11179–11189.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Emiel Hoogeboom, Alexey A. Gritsenko, Jasmijn Bastings, Ben Poole, Rianne van den Berg, and Tim Salimans. 2022. Autoregressive diffusion models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. 2021. Argmax flows and multinomial diffusion: Towards non-autoregressive language models. *CoRR*, abs/2102.05379.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing , EMNLP 2004, A meeting of SIGDAT, a Special Interest Group of the ACL, held in conjunction with ACL 2004, 25-26 July 2004, Barcelona, Spain*, pages 388–395. ACL.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 3045–3059. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

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

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. 2022. Dpm-solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. *CoRR*, abs/2206.00927.

Alexander Quinn Nichol and Prafulla Dhariwal. 2021. Improved denoising diffusion probabilistic models. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 8162–8171. PMLR.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *SCIENCE CHINA Technological Sciences*.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

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. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical text-conditional image generation with CLIP latents. *CoRR*, abs/2204.06125.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 10674–10685. IEEE.

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. *CoRR*, abs/2205.11487.

Chitwan Saharia, William Chan, Saurabh Saxena, and Mohammad Norouzi. 2020. Non-autoregressive machine translation with latent alignments. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 1098–1108. Association for Computational Linguistics.

Nikolay Savinov, Junyoung Chung, Mikolaj Binkowski, Erich Elsen, and Aäron van den Oord. 2022. Step-unrolled denoising autoencoders for text generation. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 2256–2265. JMLR.org.

Jiaming Song, Chenlin Meng, and Stefano Ermon. 2021. Denoising diffusion implicit models. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Tianxiang Sun, Yunfan Shao, Hong Qian, Xuanjing Huang, and Xipeng Qiu. 2022. Black-box tuning for language-model-as-a-service. In *Proceedings of the 39th International Conference on Machine Learning, ICML 2022, Baltimore, Maryland, USA*.

Alex Wang and Kyunghyun Cho. 2019. BERT has a mouth, and it must speak: BERT as a markov random field language model. *CoRR*, abs/1902.04094.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, pages 1097–1100. ACM.

A Generation Process

During sampling phase, all three generative masked language models show an easy-first generative behavior through time. Table 3 demonstrates an example in unconditional settings. When the context is extremely sparse, the model tends to generate tokens that is most frequently appearing (and is least surprising) in the training corpus to achieve a higher likelihood, though the generated sequences exhibit no consistency. As the context becomes richer, more details come up in the sequence. This phenomenon indicates the discrepancy between training and sampling: absorbing discrete diffusion models for text generation corrupt all tokens independently with the same noise schedule during training, while the neural network prefers tokens that are less informative when most of the input is masked.

B Templates of Conditional Generation

We show in Table 4 how the source sentences in `Seq2seq` tasks are transformed into our input template. Instead of simply concatenating source and target text, such format offers DiffusionBERT with more information to generate the desired outputs.

C Length of Generated Text

NAR methods have long been faced with the problem of fixed length generation. Unlike their AR counterparts, NAR methods are generally not able to dynamically determine the end of the sequence during generation. Especially in constrained settings, length of the optimal output depends greatly on the source sentence and cannot be assigned in advance. Existing solutions include length prediction modules (Gu et al., 2019) and generating `[PAD]` tokens (Gong et al., 2022). DiffusionBERT adopts the latter method for target length determination. In particular, we set the overall sequence length to 128 and train DiffusionBERT to predict all `[MASK]` tokens according to the instruction. Predictions consist of the generated target text followed by a `[SEP]` token and a series of `[PAD]` tokens. As shown in Table 4, the target length is dynamic depending on the position of `[SEP]`.

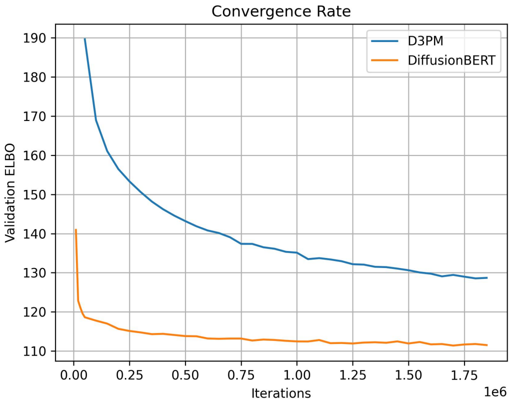


Figure 4: Curve of validation ELBO during training.

D Training Speed

Thanks to *Time Agnostic Decoding*, we introduce no additional parameters into our backbone. Thus training DiffusionBERT equals to finetuning BERT to generate text, which is relatively easier than training from scratch. In unconditional training, DiffusionBERT converges remarkably faster than D3PM. Even if the training budget is cut to 30% that of D3PM, DiffusionBERT is still able to match the performance reported in Table 1. Figure 4 demonstrates the curve of validation ELBO in the training process. Such superiority in convergence speed also holds in constrained settings. Besides, with the help of PLMs, DiffusionBERT can be well trained with smaller batch size and requires less computational resources. Though trained with half the number of GPU cores, DiffusionBERT achieves a 80% convergence acceleration compared to DiffuSeq on the QQP dataset.

BERT-Mouth	$t = 0$	`[MASK] [MASK] [MASK] [MASK] [MASK] [MASK] [MASK] [MASK] [MASK] [MASK]`
	$t = 8$	`[MASK]` **`of`** `[MASK]` **`five`** `[MASK]` **`remain`** `[MASK]` **`in`** `[MASK]` **`.`**
	$t = 16$	**`two of`** `[MASK]` **`five structures remain`** `[MASK]` **`this location .`**
	$t = 24$	**`five of`** `[MASK]` **`the windows remain at this location .`**
	$t = 32$	**`most of even the windows stand still this day .`**
D3PM	$t = 0$	`[MASK] [MASK] [MASK] [MASK] [MASK] [MASK] [MASK] [MASK] [MASK] [MASK]`
	$t = 8$	`[MASK] [MASK] [MASK] [MASK]` **`been`** `[MASK] [MASK] [MASK] [MASK]` **`.`**
	$t = 16$	`[MASK] [MASK] [MASK] [MASK]` **`been`** `[MASK] [MASK]` **`the`** `[MASK]` **`.`**
	$t = 24$	`[MASK] [MASK] [MASK]` **`also been`** `[MASK]` **`by the`** `[MASK]` **`.`**
	$t = 32$	**`the man has also been arrested by the police .`**
DiffusionBERT	$t = 0$	`[MASK] [MASK] [MASK] [MASK] [MASK] [MASK] [MASK] [MASK] [MASK] [MASK]`
	$t = 8$	`[MASK]` **`,`** `[MASK] [MASK] [MASK] [MASK] [MASK]` **`that`** `[MASK]` **`.`**
	$t = 16$	**`today ,`** `[MASK]` **`will be`** `[MASK] [MASK]` **`that`** `[MASK]` **`.`**
	$t = 24$	**`today ,`** `[MASK]` **`will be remembered for that mistake .`**
	$t = 32$	**`today , he will be remembered for that mistake .`**

Table 3: Examples generated by three generative masked language models. All three models yield words of higher frequency when t is small and tend to generate more informative tokens as the reverse diffusion process goes on.

Task	**Question Generation**	**Paraphrase**
Template	Answer: <src>. Question: <tgt>	The sentence "<src>" is equal to "<tgt>"
Input Example	Answer: She's into video games. Question: `M M M M M M M M`	The sentence "Ava feels happy." is equal to "`M M M M M M M M`
Generation Examples	What is Ava doing now? `[SEP] [PAD]` Is Ava free now? `[SEP] [PAD] [PAD]`	Ava feels positive. `[SEP] [PAD] [PAD] [PAD]` Ava is in a cheerful state. `[SEP]`

Table 4: Instruction templates and their corresponding generation examples in conditional generation. <src> and <tgt> refers to the source and target text in one data sample, respectively. `M` is the abbreviation of `[MASK]` token.

Method	**Steps**	**Inference Time (secs)**	**PPL**
DiffusionBERT	2	0.66	313.57
	8	1.39	91.01
	16	1.80	75.66
	64	4.25	65.83
	128	7.53	63.78
	512	27.48	54.63
Diffusion-LM	2000	83.67	112.12
BERT-Mouth	64	2.18	142.89
	512	14.39	86.78
GPT	64	1.55	38.7

Table 5: Comparison of inference time and perplexity among baselines and DiffusionBERTin unconditional generation.

E Sampling Speed

With the x_0-parameterization proposed in Song et al. (2021) and Austin et al. (2021), DiffusionBERT is able to perform inference with any given budget by controlling the step size in the reverse process. We also control the sampling time of BERT-Mouth by adjusting the max iteration count of its mask-predict process. We list the decoding speed and the corresponding perplexity on the LM1B test set in Table 5. Overall, DiffusionBERT exhibits competitive performance even when it reaches comparable speed to GPT and outperforms BERT-Mouth in efficiency-performance tradeoff.

Recent works have proposed some higher-order ODE solvers to accelerate diffusion models in continuous domain (Lu et al., 2022). By leveraging the all-tokens-in-one-forward nature of NAR text generation, we look forward to discovering the potential of DiffusionBERT in sampling speed. Specifically, the generation time of AR models is limited by the sequence length n. While DiffusionBERT decomposes the sampling process through time t, which can be optimized by advanced diffusion model sampler so that t is much smaller than n. We leave this for future work.

F Implementation of Evaluation Metrics

We evaluate the performance of DiffusionBERT with 3 n-gram-based methods and Rouge-L. The n-gram methods, namely BLEU-4, Self-BLEU-4 and Div-4, are implemented based on `NLTK`. We use the implementation in `torchmetrics` to compute Rouge-L scores.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
The limitations are discussed in the first section after the conclusion.

☑ A2. Did you discuss any potential risks of your work?
The potential risks are discussed in the second section after the conclusion.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Abstract and 1. Introduction

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

4.1 Tasks and Datasets, 4.2 Baselines

☑ B1. Did you cite the creators of artifacts you used?
4.1 Tasks and Datasets, 4.2 Baselines

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
All the datasets used in the submission (listed in 4.1 Tasks and Datasets) are publicly accessible for research use

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
4.1 Tasks and Datasets

☑ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
In the Ethics Statement section.

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
4.1 Tasks and Datasets

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
4.1 Tasks and Datasets

C ☑ Did you run computational experiments?

4 Experiments and Appendix D

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
4.3 Experimental Setup and Appendix D

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
4.3 Experimental Setup

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
4.4 Unconditional Generation

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Appendix F Implementation of Evaluation Metrics

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

□ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

□ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

□ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

□ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.