Few-shot Temporal Pruning Accelerates Diffusion Models for Text Generation

Bocheng Li[1,3], **Zhujin Gao**[1,3], **Yongxin Zhu**[2,3], **Kun Yin**[4], **Haoyu Cao**[4]
Deqiang Jiang[4], **Linli Xu**[1,3 *]
[1]School of Computer Science and Technology, University of Science and Technology of China
[2]School of Data Science, University of Science and Technology of China
[3]State Key Laboratory of Cognitive Intelligence
[4]Tencent YouTu Lab
{bcli,gaozhujin,zyx2016}@mail.ustc.edu.cn
{zhanyin,rechycao,dqiangjiang}@tencent.com, linlixu@ustc.edu.cn

Abstract

Diffusion models have achieved significant success in computer vision and shown immense potential in natural language processing applications, particularly for text generation tasks. However, generating high-quality text using these models often necessitates thousands of iterations, leading to slow sampling rates. Existing acceleration methods either neglect the importance of the distribution of sampling steps, resulting in compromised performance with smaller number of iterations, or require additional training, introducing considerable computational overheads. In this paper, we present Few-shot Temporal Pruning, a novel technique designed to accelerate diffusion models for text generation without supplementary training while effectively leveraging limited data. Employing a Bayesian optimization approach, our method effectively eliminates redundant sampling steps during the sampling process, thereby enhancing the generation speed. A comprehensive evaluation of discrete and continuous diffusion models across various tasks, including machine translation, question generation, and paraphrasing, reveals that our approach achieves competitive performance even with minimal sampling steps after down to less than 1 minute of optimization, yielding a significant acceleration of up to 400x in text generation tasks.

Keywords: Natural Language Generation, Machine Translation, Statistical and Machine Learning Methods

1. Introduction

Diffusion models have achieved remarkable performance across various application domains, including computer vision and natural science (Dhariwal and Nichol, 2021; Huang et al., 2022). They have also drawn considerable interests for the potential in natural language processing applications, particularly text generation tasks (Li et al., 2022b; Gong et al., 2022; Gao et al., 2022), where diffusion models offer numerous advantages such as generating high-quality, diverse, and controllable text and providing resilience against common generation issues like mode collapse (Rombach et al., 2022). Additionally, they enable manipulation and transfer of text attributes, such as style and content, which is of great significance for various problems such as open-ended NLG tasks.

However, the practical application of diffusion models in text generation has been restricted due to their inherently slow sampling processes, which requires numerous iterations to generate high-quality samples. Consequently, researchers have developed a variety of acceleration techniques for the sampling processes of diffusion models. Most of these methods either ignore the significance of the distribution of sampling steps (Song et al., 2020), causing poor performance with fewer iterations, or demand extensive data processing and extra training (Luhman and Luhman, 2021), incur-

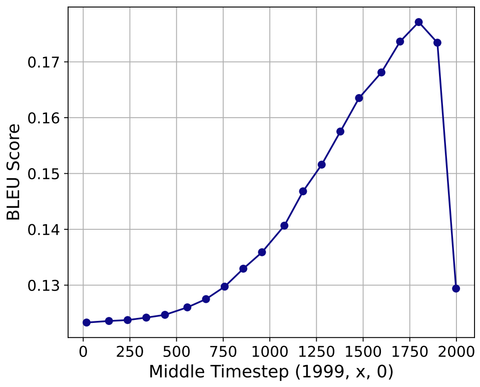

Figure 1: The contribution of individual sampling steps to the overall diffusion sampling process varies significantly. In the 3-step sampling process for the paraphasing task in the DiffuSeq model trained with 2000 diffusion steps, fixing the initial and final sampling steps and varying the middle step leads to substantial changes in the BLEU scores of the generated samples. Here every point indicate a 3-step sampling process using (1999,x,0) steps.

* Corresponding author

ring higher computational cost and complexity. As a consequence, these methods may not be applicable under tight resource and time restrictions.

Our analysis reveals that individual sampling steps have varying contributions to the overall sampling process. As illustrated in Figure 1, when using a 3-step sampling procedure for the QQP dataset of the paraphrasing task, if we fix the initial and final steps and adjust the middle step, significant fluctuations in the BLEU scores of the generated samples are witnessed. This observation highlights the potential for identifying the optimal sampling steps, with the aim of reducing the number of sampling steps while maintaining the performance of diffusion models.

In this paper, in order to accelerate the sampling process while maintaining the model's performance, we aim to prune the sampling steps that contribute minimally to the overall sampling process. To achieve that, we propose Few-shot Temporal Pruning, an innovative method designed to effectively leverage limited data to significantly accelerate diffusion models in text generation tasks.

Our approach starts with discriminating the "significant steps" and "redundant steps" based on their influences on the model performance during the sampling process. Subsequently, we employ an efficiently configured Bayesian optimizer to iteratively identify the significant steps by evaluating the model performance based on various pruned sampling steps in a few-shot setting. This process enables us to select the significant steps and remove the redundant ones accordingly throughout the entire sampling process. As a result, our method can attain improved performance with significantly fewer number of sampling steps, effectively speeding up the sampling process.

In addition, a distinguishing feature of Few-shot Temporal Pruning is its ability to effectively utilize limited data in a few-shot manner, eliminating the requirement for additional training or extensive data processing. Specifically, our method is capable of significantly improving the model performance using just 20 items in the validation set. This makes it suitable for scenarios with limited computational resources or data availability, such as real-time text generation applications or tasks requiring rapid adaptation to new data.

The effectiveness of Few-shot Temporal Pruning is validated through comprehensive experiments comparing to traditional diffusion models, where our approach achieves comparable performance with only 4 sampling steps instead of the 50-2000 steps required by the baselines, resulting in a significant acceleration for various text generation tasks.

Our contributions in this paper are as follows:

- We introduce Few-shot Temporal Pruning, a novel method designed to accelerate diffusion models in text generation by pruning redundant sampling steps utilizing an efficient Bayesian optimization approach in a few-shot manner.

- We perform a thorough qualitative analysis of the effects of redundant sampling steps on the model performance and the optimized distribution of sampling steps. Our analysis reveals two key findings: 1) redundant steps may hinder the model's capacity to make further modifications to sentences, and 2) diffusion models are subject to insufficient noise exposure during the early sampling steps.

- Through a comprehensive experimental investigation, we demonstrate the effectiveness of Few-shot Temporal Pruning in comparison to traditional diffusion models, achieving a substantial speed-up for a wide range of text generation tasks.

2. Related Work

2.1. Diffusion Models

The diffusion model is derived by sampling from the inverse of a noise-increasing process, which starts with a simple noise x_T and proceeds to generate a series of cleaner samples by reducing the noise at each time step t.

We define the initial data distribution $x_0 \sim q(x_0)$ and a Markovian process that produces $x_1, \ldots, x_T$ by adding Gaussian noise controlled by β_t:

$$q(x_t \mid x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t} x_{t-1}, \beta_t \mathbf{I}) \quad (1)$$

where $\beta_t = g(t)$, and $g(\cdot)$ can be seen as the noise schedule of the diffusion model.

We can interpret $q(x_t \mid x_0)$ as a Gaussian distribution with a reparameterization trick:

$$\begin{aligned} q(x_t \mid x_0) &= \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t} x_0, (1-\bar{\alpha}_t)\mathbf{I}) \\ &= \sqrt{\bar{\alpha}_t} x_0 + \epsilon \sqrt{1-\bar{\alpha}_t}, \end{aligned} \quad (2)$$

where $\epsilon \sim \mathcal{N}(0, \mathbf{I}), \alpha_t := 1 - \beta_t, \bar{\alpha}_t := \prod_{s=1}^{t} \alpha_s$.

The posterior distribution $q(x_{t-1} \mid x_t, x_0)$ is a Gaussian distribution with mean $\tilde{\mu}_t(x_t, x_0)$ and variance $\tilde{\beta}_t$:

$$q(x_{t-1} \mid x_t, x_0) = \mathcal{N}(x_{t-1}; \tilde{\mu}(x_t, x_0), \tilde{\beta}_t \mathbf{I}) \quad (3)$$

To sample from $q(x_0)$, we start by sampling from $q(x_T)$ and then sample to x_0 using the inverse steps $q(x_{t-1} \mid x_t)$. A neural network is used to approximate $q(x_{t-1} \mid x_t)$:

$$p_\theta(x_{t-1} \mid x_t) = \mathcal{N}(x_{t-1}; \tilde{\mu}(x_t, \hat{x}_0), \tilde{\beta}_t \mathbf{I}), \quad (4)$$

where $\hat{x}_0$ is the prediction of the model, which is trained to learn the true data distribution $q(x_0)$ by optimizing the following variational lower bound L_{vlb} for $p_\theta(x_0)$:

$$\begin{aligned} L_{\text{vlb}} &= L_0 + L_1 + \ldots + L_{T-1} + L_T \\ L_0 &= -\log p_\theta\left(x_0 \mid x_1\right) \\ L_{t-1} &= D_{KL}\left(q\left(x_{t-1} \mid x_t, x_0\right) \| \, p_\theta\left(x_{t-1} \mid x_t\right)\right) \\ L_T &= D_{KL}\left(q\left(x_T \mid x_0\right) \| \, p\left(x_T\right)\right) \end{aligned} \tag{5}$$

2.2. Text Generation with Diffusion Models

Diffusion models have achieved significant success in computer vision (Ho et al., 2020; Rombach et al., 2022), prompting extensive investigation into their potential application to text generation tasks. However, applying diffusion models to text is challenging due to the discrete nature of textual data. Several approaches have been developed to address this challenge to retain the advantages of diffusion models, such as high-quality, diversity and controllable text generation, while offering fresh insights into understanding and modeling the complex structures of natural language. Representatives include the Multinomial Diffusion Model (Hoogeboom et al., 2021), D3PM (Austin et al., 2021a), Bit Diffusion (Chen et al., 2022), latent diffusion models (Yu et al., 2022), step-wise text generation with SUNDAE (Savinov et al., 2021), and edit-based text generation with DiffusER (Reid et al., 2022). Another line of research leverages word element embeddings to allow for continuous diffusion models to be applied to discrete text, with approaches like Diffusion-LM (Li et al., 2022a).

2.3. Acceleration of Diffusion Models

The computational efficiency of diffusion models remains a limiting factor in their extensive research and applications. Researchers have proposed various methods to accelerate the sampling processes of diffusion models, such as SDE Solvers (Song and Ermon, 2019; Dockhorn et al., 2022), ODE Solvers (Song et al., 2020; Liu et al., 2022), the truncated diffusion process (Lyu et al., 2022), and knowledge distillation (Luhman and Luhman, 2021; Salimans and Ho, 2022). Recently, a sampling strategy employing a crafted step distribution is proposed in (Tang et al., 2023) to achieve better sample results compared to using a uniform step distribution.

Despite these developments, further reduction of sampling iterations in high-performance text generation remains a challenge. Current diffusion models are often slower than traditional auto-regressive language models when generating text, due to the large number of sampling iterations required at runtime, as demonstrated by DiffuSeq (Gong et al., 2022).

3. Temporal Pruning

The idea of Temporal Pruning is motivated by the observation that when the number of sampling steps in the diffusion model is limited, the model performance does not deteriorate significantly when optimal steps are chosen for sampling. This indicates that most of the sampling steps are redundant, which can be pruned to accelerate the sampling process without impairing the results.

We further define the redundant and significant steps as follows:

Redundant Steps are the sampling steps at which the generated samples do not exhibit significant changes from the previous samples, thus having minimal influence on the final generated output. Pruning these steps enhances the efficiency of the sampling process without compromising the quality.

Significant Steps correspond to the points where samples experience substantial changes compared to the previous ones. By eliminating these samping steps, the progressive refinement of samples would be disrupted, and diffusion models would no longer generate high-quality outputs.

Building upon this idea, we propose the scheme of temporal pruning via optimization algorithms. We provide an overview of our method in Figure 2.

For the sake of simplicity, we consider the entire frozen diffusion model as a black box. Our primary objective is to optimize its input, which is the distribution of sampling steps. This distribution is represented as a list of individual sampling steps, formally defined as $\mathbf{S} = [S_1, \ldots, S_{N_{\text{pruned}}}]$, where each S_i in $\mathbf{S}$ denotes a distinct sampling step, with i ranging from 1 to the number of sampling steps after pruning, N_{pruned}.

To identify the redundant steps and significant steps according to their influence on the model's performance, we employ various distributions of sampling steps to generate the sample output $\hat{x}_0 = M(\mathbf{S})$, where $M(\cdot)$ represents the diffusion model. This output is subsequently evaluated against the ground truth, denoted by Y_{gt}. In our context, we utilize the BLEU score, represented as $\text{BLEU}(M(\mathbf{S}), Y_{\text{gt}})$, to assess the model's overall performance. Consequently, we define our performance function as $R(\cdot) = \text{BLEU}(M(\cdot), Y_{\text{gt}})$.

To find the optimal distribution of sampling steps $\mathbf{S}_{\text{optimal}}$ that maximizes the performance function $R(\cdot)$ within the entire set of possible distributions, referred to as domain $\mathcal{D}$, one might consider a brute-force approach that computes all the $R(\mathbf{S})$

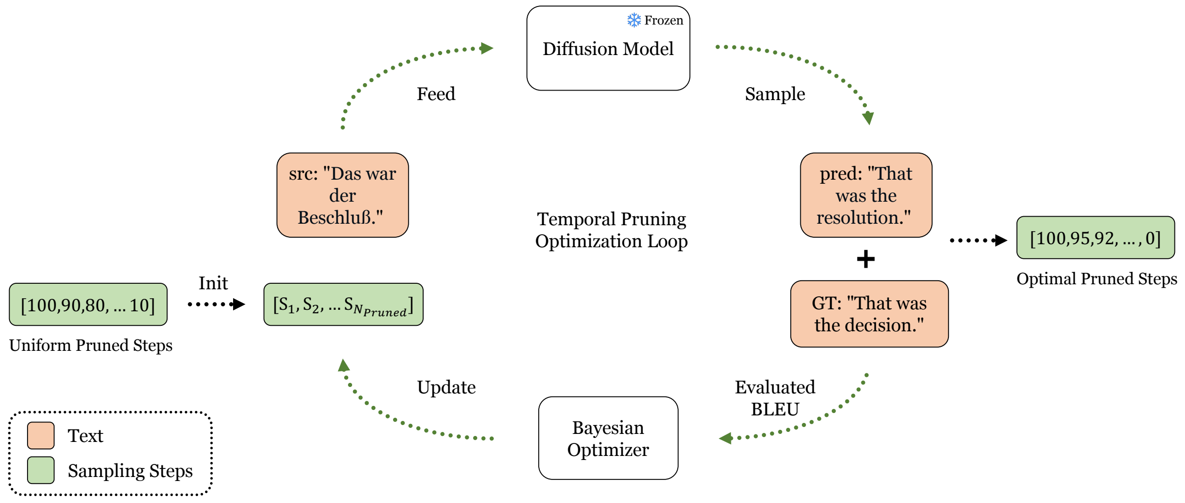

Figure 2: The Temporal Pruning method involves: (1) Uniformly initializing the pruned sampling steps, (2) generating predictions using the initial sampling steps and source text in the validation set, (3) evaluating these predictions with BLEU, (4) integrating BLEU scores and sampling steps into the Bayesian optimization framework, (5) leveraging the prior information to estimate and update the pruned steps, (6) iteratively refining the pruned sampling steps to improve sample quality and (7) obtaining the optimal distribution of sampling steps.

values, where $\mathbf{S} \in \mathcal{D}$. However, this method becomes computationally inefficient for diffusion models with a large maximum number of sampling steps (e.g. Gong et al. (2022)), while exhaustively enumerating and evaluating all distributions of sampling steps requires an exponential number of optimizations, which is prohibitive in practice.

3.1. Temporal Pruning via Bayesian Optimization

Given the high computational cost of each sampling process, we adopt a Bayesian optimization approach specifically designed for optimizing the black-box functions. This method enables the efficient identification of significant steps without the need to iterate through the entire domain $\mathcal{D}$.

Algorithm 1 presents the complete pseudo-code for Temporal Pruning with Bayesian Optimization. The method utilizes an observation set O, which stores all explored distributions of sampling steps $\mathbf{S}$ along with their corresponding performance values, denoted by $R(\mathbf{S})$. By incorporating this prior information, the Bayesian optimizer is able to guide the selection of the next distribution $\mathbf{S}_{\text{next}}$ for more effective evaluation.

The initialization starts with the selection of the desired number of pruned steps, represented by N_{pruned}. Following this, the pruned sampling steps $\mathbf{S}_{\text{init}}$ are uniformly initialized.

After initializing $\mathbf{S}_{\text{init}}$ and calculating the corresponding performance function $R(\mathbf{S})$, we store the pair $(\mathbf{S}, R(\mathbf{S}))$ in the observation set O, and update the Gaussian process (GP) posterior, which models the objective function (in our context, also the performance function) $R(\cdot)$, and then incorporate it into a Gaussian process regression framework.

To determine the next $\mathbf{S}$ to explore, we rely on the Gaussian process (GP) posterior. Instead of

Algorithm 1: Temporal Pruning via Bayesian Optimization

```
Input: Frozen Diffusion Model $M$,
  Number of Pruned Steps $N_{\text{pruned}}$,
  Optimization iterations $n_{\text{iter}}$,
  Domain $\mathcal{D}$,
  Ground truth $Y_{\text{gt}}$
Output: Optimal Sampling Steps $\mathbf{S}_{\text{optimal}}$
1 Uniformly initialize the list of pruned steps
    $\mathbf{S}_{\text{init}}$
2 Compute $R(\mathbf{S}_{\text{init}})$ and initialize the
    observation set $O \leftarrow \{(\mathbf{S}_{\text{init}}, R(\mathbf{S}_{\text{init}}))\}$
3 for $i = 1$ to $n_{\text{iter}}$ do
4     Update the GP posterior given
        observations $O$
5     Pick a subset $\mathcal{D}'$ of $\mathcal{D}$ using 20 iterations
        of Limited-memory BFGS with 5
        random initial points, compute
        $\alpha_{\text{GP-Hedge}}(\mathbf{S})$ for all $\mathbf{S}$ in $\mathcal{D}'$
6     Select the next observation point
        $\mathbf{S}_{\text{next}} = \arg\max_{\mathbf{S} \in \mathcal{D}'} \alpha_{\text{GP-Hedge}}(\mathbf{S})$
7     Compute $R(\mathbf{S}_{\text{next}})$ and update the
        observation set
        $O = O \cup \{(\mathbf{S}_{\text{next}}, R(\mathbf{S}_{\text{next}}))\}$
8 end
9 $\mathbf{S}_{\text{optimal}} = \arg\max_{\mathbf{S} \in O} R(\mathbf{S})$
```

performing a costly sampling process for a subset $\mathcal{D}'$ of the domain $\mathcal{D}$, we compute an inexpensive acquisition function $\alpha(\cdot)$. This acquisition function is utilized to identify the most promising $\mathbf{S}$ for exploration. Initially, the subset $\mathcal{D}'$ is chosen randomly. Subsequently, we employ the limited-memory BFGS optimization algorithm (Liu and Nocedal, 1989) to further probe the local maxima within the subset using $\alpha(\cdot)$. The optimal local maximum is selected as the next $\mathbf{S}$ to explore $\mathbf{S}_{\text{next}} = \arg\max_{\mathbf{S}\in\mathcal{D}'} \alpha(\mathbf{S})$.

The process of computing the next pair $(\mathbf{S}, R(\mathbf{S}))$ and updating the observation set O as well as the GP posterior is iteratively performed to further explore the observation set. After a predetermined number of optimization iterations n_{iter}, we obtain the optimal distribution of sampling steps $\mathbf{S}_{\text{optimal}} = \arg\max_{\mathbf{S}\in O} R(\mathbf{S})$.

In our approach, we employ three distinct acquisition functions: Lower Confidence Bound (LCB), Negative Expected Improvement (NEI), and Negative Probability of Improvement (NPI) (Brochu et al., 2011). To adaptively choose the optimal acquisition function from them, we utilize the portfolio-based GP-Hedge algorithm (Brochu et al., 2011).

To further accelerate the searching process, we employ a strategy that involves fixing the first sampling step, S_1, to the maximum number of diffusion steps in the model. However, the last sampling step, $S_{N_{\text{pruned}}}$, is unfixed, enabling the optimizer to find the best terminate step, preserving high-quality results before the performance deteriorates.

3.2. Few-shot Temporal Pruning

During each iteration of the temporal pruning optimization process, we perform a complete sampling process to calculate the performance function $R(\cdot) = \text{BLEU}(M(\cdot), Y_{\text{gt}})$. The amount of data used for sampling directly affects the speed and computational cost of R.

To demonstrate the robustness of our method in handling limited data, we introduce a few-shot variant of temporal pruning.

The main optimization loop of this approach remains consistent with the original method. However, the data used for calculating the performance function $R(\cdot)$ is limited to a randomly selected subset of items from the validation set. This strategy prunes redundant sampling steps and enhances efficiency without requiring a large amount of data.

By leveraging the few-shot samples, the optimizer is still capable of identifying the significant sampling steps that preserve the model's overall performance. This few-shot variant of temporal pruning demonstrates the versatility of our method in tackling scenarios with limited data availability while maintaining the high quality of the results.

3.3. Computational Cost

The computation cost of each Temporal Pruning iteration involves: (1) selecting the next $\mathbf{S}$ from the observation set O comprised of $(\mathbf{S}, R(\mathbf{S}))$ pairs, and (2) calculating $R(\mathbf{S})$.

The time complexity in (1) primarily consists of:

1. The time complexity for updating the GP posterior with observations O is $O(n^3)$, where n is the number of training samples, capped at n_{iter}.
2. The selection of the subset $\mathcal{D}'$ from $\mathcal{D}$, using 20 iterations of Limited-memory BFGS with 5 initial points, calculates $\alpha_{\text{GP-Hedge}}(\mathbf{S})$ for all $\mathbf{S}$ in $\mathcal{D}'$. This process includes the optimization of EI, LCB, and PI acquisition functions, resulting in an overall time complexity of $O(N_{\text{pruned}})$.
3. Selecting the next observation point $\mathbf{S}_{\text{next}} = \arg\max_{\mathbf{S}\in\mathcal{D}'} \alpha_{\text{GP-Hedge}}(\mathbf{S})$ is accomplished in constant time, $O(1)$, given the fixed number of iterations and initial points.

The time complexity in (2) is primarily due to the computation of $R(\mathbf{S}_{\text{next}})$ and the update of the observation set, which approximately introduce a time complexity of $O(L_{\text{source}})$, with L_{source} denoting the size of the selected source text in the validation set. This step constitutes the primary computational overhead of temporal pruning.

4. Experimental Settings

4.1. Datasets and Tasks

We employ five benchmark datasets across various tasks:

Machine Translation. We adopt IWSLT14 DE-EN (Cettolo et al., 2014), WMT16 EN-RO (Bojar et al., 2016) and WMT14 EN-DE (Bojar et al., 2014) which contain 160K/7K/7K, 610K/2K/2K and 4.0M/3K/3K training/validation/testing pairs respectively.

Question Generation and Paraphrasing. We use Quasar-T for Question Generation (QG) (Dhingra et al., 2017) and Quora Question Pairs (QQP) for Paraphrasing, which contain 117K/2K/10K and 145K/2K/2.5K training/validation/testing pairs respectively.

For all the machine translation tasks, we work with the original data and do not use knowledge distillation. For the QG and Paraphasing tasks, we follow (Gong et al., 2022) and use their preprocessed data splits, and use BERT (Devlin et al., 2019)'s WordPiece tokenization (30522 vocabulary). We report BLEU (Papineni et al., 2002) scores for all the tasks.

Model	Steps	Shots	IWSLT14 DE-EN	WMT16 EN-RO	WMT14 EN-DE	QG	QQP
Transformer-base	-	-	34.51§	34.16§	27.53§	16.63†	27.22†
Absorbing Diffusion							
Vanilla	50	-	28.95	30.88	22.98	17.49	24.34
Vanilla	4	-	27.16	27.41	18.70	17.45	24.07
Temporal Pruning	4	Full Set	28.61	31.03	22.42	17.47	24.41
Temporal Pruning	4	20	28.12	29.51	21.69	17.47	24.21
Multinomial Diffusion							
Vanilla	50	-	13.12	4.50	0.32	17.45	24.06
Vanilla	4	-	24.23	27.80	17.19	17.08	21.52
Temporal Pruning	4	Full Set	26.96	29.69	21.44	17.48	23.70
Temporal Pruning	4	20	26.83	29.88	20.98	17.38	23.22
DiffuSeq							
Vanilla	2000	-				17.31	24.13
Vanilla	4	-	-	-	-	16.06	19.05
Temporal Pruning	4	Full Set				16.38	22.32
Temporal Pruning	4	10				16.38	21.90

Table 1: Comparison of BLEU scores for various diffusion models on text generation, with and without Temporal Pruning, assessed across multiple tasks. Notably, DiffuSeq's applied tasks (Gong et al., 2022) overlap with the other two discrete diffusion models only in QG and QQP. The term "Shots" refers to the number of items from the validation set of the corresponding dataset used when implementing Temporal Pruning. The term "Vanilla" denotes uniform sampling steps taken from the original implementations. † indicates the results from Gong et al. (2022), and § indicates the results from Zheng et al. (2023).

4.2. Baselines

Our baseline models fall into three categories: autoregressive, discrete diffusion, and continuous diffusion models.

For the autoregressive models, we choose Transformer (Vaswani et al. (2017)) as a strong baseline.

For the discrete diffusion models, we consider two strong baselines: Absorbing Diffusion (Austin et al., 2021b) and Multinomial Diffusion (Hoogeboom et al., 2021), both with vanilla uniform sampling steps.

For the continuous diffusion models, we consider DiffuSeq (Gong et al., 2022) with vanilla uniform sampling steps. DiffuSeq uses DDIM (Song et al. (2020)) sampling when the number of timesteps is smaller than the original training steps (2000).

Specifically, "vanilla" here refers to the uniform step distribution that corresponds with the original training steps of each model, as inherited from their initial implementations. For example, we use $\mathbf{S} = [1999, 1333, 666, 0]$ for DiffuSeq's 4-step sampling process with 2000 training steps.

4.3. Benchmarking Temporal Pruning

We conduct a thorough assessment of Temporal Pruning, examining the running time required by Temporal Pruning and inference speedup of models with Temporal Pruning.

Running Time of Temporal Pruning. We assess the computational cost of Temporal Pruning by timing its two components (Section 3.3) and varying the data volume, referred to as "shots", for sampling across Temporal Pruning variants.

Inference Speedup. In our experiments, baseline models use their maximum training steps (2000 for DiffuSeq, 50 for Absorbing and Multinomial Diffusion) for sampling, while models with Temporal Pruning use 4 steps ($N_{\text{pruned}} = 4$), facilitating a direct analysis of the sampling speed.

4.4. Implementation Details

For the discrete diffusion (Multinomial Diffusion and Absorbing Diffusion) baselines, we follow the sampling methods in Zheng et al. (2023). For continuous diffusion models, we adopt the implementation provided in Gong et al. (2022).

To constrain the search space and minimize unnecessary computation, we set the maximum sampling steps to 50 and 2000, which are the training steps for discrete and continuous diffusion models, respectively. For all Temporal Pruning variants, we set n_{iter} to 50. Few-shot tasks with discrete diffusion models use 20 randomly selected items in the subset, while continuous diffusion models use 10. In contrast, full-set Temporal Pruning employs the entire validation sets for the corresponding tasks.

When analyzing the the running time of Temporal Pruning and inference speedups, we follow the testing conditions from Zheng et al. (2023), using a batch size of 32 on an NVIDIA RTX3090.

Model	Shots	Total	Inference	Bayesian
DiffuSeq				
QQP	Full	19.22 min	19.22 min	0.04 s
	20	0.63 min	0.63 min	0.04 s
QG	Full	18.94 min	18.94 min	0.04 s
	20	0.66 min	0.66 min	0.04 s
Absorbing				
IWSLT14 DE-EN	Full	69.17 min	69.17 min	0.04 s
	20	0.76 min	0.76 min	0.04 s
WMT16 EN-RO	Full	22.50 min	22.50 min	0.04 s
	20	0.84 min	0.84 min	0.04 s
WMT14 EN-DE	Full	33.33 min	33.33 min	0.04 s
	20	0.84 min	0.84 min	0.05 s
QG	Full	50.00 min	50.00 min	0.04 s
	20	0.78 min	0.78 min	0.05 s
QQP	Full	45.83 min	45.83 min	0.04 s
	20	0.82 min	0.82 min	0.05 s
Multinomial				
IWSLT14 DE-EN	Full	80.00 min	80.00 min	0.04 s
	20	0.92 min	0.92 min	0.04 s
WMT16 EN-RO	Full	36.67 min	36.67 min	0.04 s
	20	0.96 min	0.96 min	0.04 s
WMT14 EN-DE	Full	51.67 min	51.67 min	0.04 s
	20	0.93 min	0.93 min	0.04 s
QG	Full	65.83 min	65.83 min	0.04 s
	20	0.90 min	0.90 min	0.04 s
QQP	Full	54.17 min	54.17 min	0.04 s
	20	0.93 min	0.93 min	0.04 s

Table 2: The running time of Temporal Pruning. "Shots" aligns with the term in Table 1. "Total" represents the overall runtime, while "Inference" and "Bayesian" denote the cumulative times for components (2) and (1) of the computation cost, as discussed in Section 3.3, respectively.

Model	TP	Baseline	Speedup
DiffuSeq			
QQP	91.54 sps	0.226 sps	405x
QG	92.14 sps	0.218 sps	423x
Absorbing			
IWSLT14 DE-EN	182.91 sps	18.50 sps	9.88x
WMT16 EN-RO	114.98 sps	11.32 sps	10.15x
WMT14 EN-DE	115.12 sps	10.95 sps	10.51x
QG	73.40 sps	6.70 sps	10.95x
QQP	92.47 sps	9.20 sps	10.05x
Multinomial			
IWSLT14 DE-EN	134.27 sps	14.51 sps	9.25x
WMT16 EN-RO	68.86 sps	6.42 sps	10.72x
WMT14 EN-DE	64.01 sps	5.92 sps	10.81x
QG	52.17 sps	4.27 sps	12.21x
QQP	80.86 sps	6.36 sps	12.71x

Table 3: Inference speed (sentence/second, denoted as "sps") for models using Temporal Pruning (with 4 steps) as compared to the baselines (with maximum sampling steps of 2000 for DiffuSeq, and 50 steps for Absorbing Diffusion and Multinomial Diffusion).

5. Results

5.1. Efficient High-Quality Text Generation with Temporal Pruning

Table 1 showcases the efficacy of Temporal Pruning across multiple tasks and metrics. Our approach substantially boosts diffusion models' performance at lower iterations using limited data while preserving text quality.

Temporal Pruning consistently improves diffusion models on machine translation tasks. For instance, Absorbing Diffusion achieves a 28.61 BLEU on IWSLT14 DE-EN with Temporal Pruning (4 steps, full validation set), surpassing the 27.16 BLEU with vanilla sampling (4 steps). Similar improvements are observed for Multinomial Diffusion and DiffuSeq. Unsurprisingly, the performance of Transformer is superior to the diffusion models due to the autoregressive nature.

In the QG and QQP tasks, Temporal Pruning also consistently enhances the performance of diffusion models with fewer iterations, as demonstrated in Table 1.

Notably, Few-shot Temporal Pruning, using only $0.27\% - 1\%$ of the validation set, achieves $95\% - 100\%$ of the BLEU score compared to using the full validation set. In the WMT16 EN-RO task, multinomial diffusion with 20-shot Temporal Pruning outperforms full-set Temporal Pruning, highlighting Temporal Pruning's effectiveness with limited data and its potential in resource-constrained scenarios.

5.2. Running Time of Temporal Pruning

Table 2 shows the remarkable computational efficiency of Temporal Pruning, often taking less than a minute in few-shot scenarios, indicating that the component (2) (referenced in Section 3.3) predominantly contributes to the overall computational cost. The variation of running time in the full-set scenarios across tasks can be attributed to the different validation set sizes, ranging from 2K to 7K entries, directly impacting the inference time. Temporal Pruning's robustness in few-shot settings allows for a reduction in the validation set size, speeding up its optimization to under a minute.

5.3. Inference Speedup

Table 3 shows the significant impact of Temporal Pruning in boosting the inference speed of various diffusion models. For DiffuSeq, the speedup surpasses 400x, accelerating the generation speed from 0.226 to a peak of 92.14 sentences/s. In both Absorbing Diffusion and Multinomial Diffusion, Temporal Pruning yields approximately a tenfold speedup, lifting the generation speed from a maximum of 18.50 to a peak of 182.91 sentences/s. These substantial improvements highlight the effectiveness of Temporal Pruning in accelerating the inference speed, thus facilitating more efficient text generation processes.

In summary, the proposed Temporal Pruning method significantly enhances the efficiency of text generation across various tasks. Its low computational cost, impressive speedup in inference, and robust performance under data constraints make

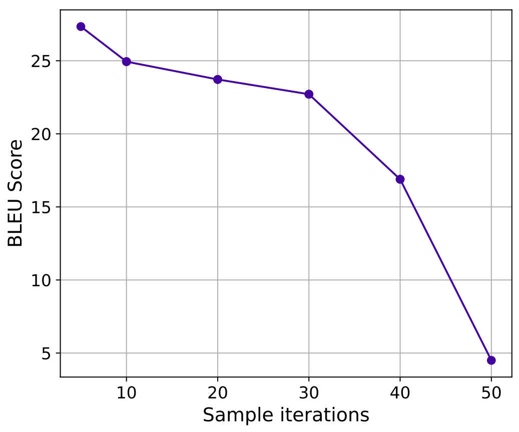

Figure 3: Performance degradation on the WMT16 test set using the vanilla multinomial diffusion model.

it a promising and efficient solution for high-quality text generation.

6. Analysis

6.1. Overcoming Sampling Degradation via Temporal Pruning

We start our analysis of Temporal Pruning with Multinomial Diffusion by examining the effect of redundant sampling steps on the model performance.

As per Zheng et al. (2023), Multinomial Diffusion, after several iterations, shows a tendency of $p_\theta(x_{t-1} \mid x_t) \approx x_t$, which essentially replicates the previous state, leading to a degradation in sampling performance. As such, it becomes crucial to attain superior results using minimal steps before the model begins to degrade. The degradation impact is shown in Figure 3, where applying vanilla multinomial diffusion to the WMT16 test set causes a significant BLEU score drop from 27.80 (4 iterations) to 4.50 (50 iterations), emphasizing the negative impact of redundant sampling steps.

Redundant steps negatively impact the model performance in two ways: initially, they may hinder sentences from reaching an optimal state, and after degradation, they consume computational resources without improving the output quality. This is observed in Table 4's upper half, where the translated sentence remains unchanged from step 0 to 50, suggesting excessive diffusion steps can hinder further sentence transformations.

By employing Temporal Pruning to eliminate redundant steps, the model can refine sentences and enhance the generation quality before its degradation. This is evident in the lower section of Table 4. Notably, despite the subsequent degradation, the overall output remains largely unaltered, confirming the efficacy of pruning redundant steps in maintaining the output quality.

Source: ich danke ihnen für ihre aufmerksamkeit.
Reference: thank you for your attention.

# Iter.	Decodes
	Vanilla Multinomial Diffusion, 50 steps
0	○ books mindestens bridge ght dahin
10	○ books mindestens bridge ght dahin
20	○ books mindestens bridge ght dahin
30	○ books mindestens bridge ght dahin
40	○ books mindestens bridge ght dahin
50	○ books mindestens bridge ght dahin
	Multinomial Diffusion with Temporal Pruning, 4 steps
0	○ jähr## dadurch sprü## vege## ined depres## de## frag##
1	○ jähr## dadurch very much for your attention .
2	○ thank you very much for your attention .
3	○ thank you very much for your attention .
4	○ thank you very much for your attention .

Table 4: A comparison of samples generated from multinomial diffusion w/ and w/o Temporal Pruning on IWSLT dataset. Words are in lower case, and ## denotes the sub-word tokenization artifacts.

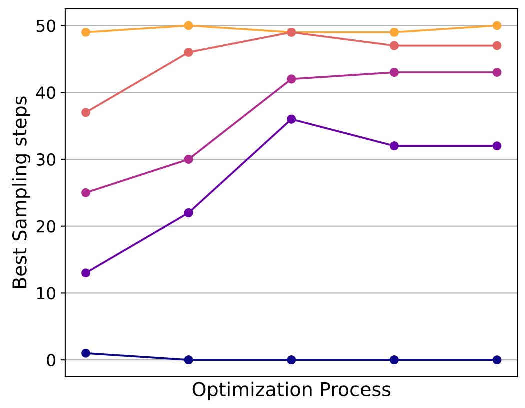

Figure 4: The distribution of sampling steps throughout the optimization process, with each color indicating an individual timestep S_i belonging to the pruned set $\mathbf{S} = [S_1, S_2, S_3, S_4, S_5]$. The colored lines from top to bottom correspond to S_1, S_2, S_3, S_4, and S_5, respectively.

Our results demonstrate that redundant steps can undermine the performance of diffusion models. These redundant steps may consume computational resources without improving the samples, blocking them from further transformation. Aggressive temporal pruning is therefore necessary for efficiently generating high-quality samples.

6.2. Insufficient Noising at Early Steps

In this study, we perform a qualitative analysis of the optimized sampling step distribution using absorbing diffusion on the WMT14 EN-DE dataset. we utilize Temporal Pruning with $N_{\text{pruned}} = 5$ to optimize the distribution of sampling steps $\mathbf{S}$, and observe its changes throughout the optimization process. The results of this analysis are illustrated in Figure 4. Initially, a uniform sampling is employed to determine the sampling steps. However, as the optimization process proceeds, we can observe a notable shift in the distribution of the optimized sampling steps, with a tendency to concentrate at higher steps. This finding suggests that higher steps are of greater importance, as the model is exposed to a substantial amount of noise during these stages, leading to more comprehensive training (Gao et al., 2022).

7. Conclusion

We present Few-shot Temporal Pruning, a robust, effective and training-free approach to accelerate diffusion models for text generation. Extensive experiments on machine translation, question generation and paraphrasing tasks demonstrate that Few-shot Temporal Pruning achieves comparable performance with significantly fewer sampling steps, resulting in a substantial speed-up for various text generation tasks. Additionally, we perform a thorough qualitative analysis of the effects of redundant sampling steps on model performance and the optimized distribution of sampling steps. Our findings reveal that redundant steps may impede the model's ability to make further modifications to sentences and diffusion models are subject to insufficient noise exposure during early sampling steps.

8. Limitations

Although Temporal Pruning can enhance the convergence rate during the sampling process, it does not significantly improve the extreme performance of the model. By modifying the distribution of sampling steps, we can generally expect faster convergence, but the gains in peak performance are limited when compared to retraining methods (e.g., reparameterization).

As a result, when the goal is to achieve better performance than what the original model offers, it is often necessary to further adjust the model structure in accordance with the text data's characteristics and retrain the model to obtain improved results. Additionally, in this paper, we do not consider the issues such as the method's applicability to languages with more complex morphology, scalability to longer texts, or the requirements of substantial computational resources. These factors should be considered in future research to ensure a comprehensive understanding of the method's limitations and potential improvements.

9. Acknowledgements

This research was supported by the National Natural Science Foundation of China (Grant No. 62276245), and Anhui Provincial Natural Science Foundation (Grant No. 2008085J31).

10. Bibliographical References

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. 2021a. Structured denoising diffusion models in discrete state-spaces. *ArXiv*, abs/2107.03006.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. 2021b. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*.

Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA. Association for Computational Linguistics.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. 2016. Findings of the 2016 conference on machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 131–198, Berlin, Germany. Association for Computational Linguistics.

Eric Brochu, Matthew W. Hoffman, and Nando de Freitas. 2011. Portfolio allocation for bayesian optimization.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2014. Report on the 11th IWSLT evaluation campaign. In *Proceedings of the 11th International Workshop on Spoken Language Translation: Evaluation Campaign*, pages 2–17, Lake Tahoe, California.

Ting Chen, Ruixiang Zhang, and Geo rey E. Hinton. 2022. Analog bits: Generating discrete data using diffusion models with self-conditioning. *ArXiv*, abs/2208.04202.

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. *ArXiv*, abs/2105.05233.

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

Tim Dockhorn, Arash Vahdat, and Karsten Kreis. 2022. Score-Based Generative Modeling with Critically-Damped Langevin Diffusion. ArXiv:2112.07068 [cs, stat].

Zhujin Gao, Junliang Guo, Xu Tan, Yongxin Zhu, Fang Zhang, Jiang Bian, and Linli Xu. 2022. Difformer: Empowering diffusion model on embedding space for text generation. *arXiv preprint arXiv:2212.09412*.

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

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. 2021. Argmax Flows and Multinomial Diffusion: Learning Categorical Distributions. In *Advances in Neural Information Processing Systems*, volume 34, pages 12454–12465. Curran Associates, Inc.

Lei Huang, Hengtong Zhang, Tingyang Xu, and Ka-Chun Wong. 2022. Mdm: Molecular diffusion model for 3d molecule generation.

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

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B Hashimoto. 2022b. Diffusion-lm improves controllable text generation. *arXiv preprint arXiv:2205.14217*.

Dong C. Liu and Jorge Nocedal. 1989. On the limited memory bfgs method for large scale optimization. *Mathematical Programming*, 45:503–528.

Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. 2022. Pseudo Numerical Methods for Diffusion Models on Manifolds. ArXiv:2202.09778 [cs, math, stat].

Eric Luhman and Troy Luhman. 2021. Knowledge Distillation in Iterative Generative Models for Improved Sampling Speed. ArXiv:2101.02388 [cs].

Zhaoyang Lyu, Xudong XU, Ceyuan Yang, Dahua Lin, and Bo Dai. 2022. Accelerating Diffusion Models via Early Stop of the Diffusion Process. ArXiv:2205.12524 [cs].

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Machel Reid, Vincent J. Hellendoorn, and Graham Neubig. 2022. Diffuser: Discrete diffusion via edit-based reconstruction. *ArXiv*, abs/2210.16886.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models.

Tim Salimans and Jonathan Ho. 2022. Progressive Distillation for Fast Sampling of Diffusion Models. ArXiv:2202.00512 [cs, stat].

Nikolay Savinov, Junyoung Chung, Mikolaj Binkowski, Erich Elsen, and Aäron van den Oord. 2021. Step-unrolled denoising autoencoders for text generation. *ArXiv*, abs/2112.06749.

Jiaming Song, Chenlin Meng, and Stefano Ermon. 2020. Denoising diffusion implicit models. In *International Conference on Learning Representations*.

Yang Song and Stefano Ermon. 2019. Generative modeling by estimating gradients of the data distribution. volume 32.

Zecheng Tang, Pinzheng Wang, Keyan Zhou, Juntao Li, Ziqiang Cao, and Min Zhang. 2023. Can diffusion model achieve better performance in text generation? bridging the gap between training and inference!

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Peiyu Yu, Sirui Xie, Xiaojian Ma, Baoxiong Jia, Bo Pang, Ruigi Gao, Yixin Zhu, Song-Chun Zhu, and Ying Nian Wu. 2022. Latent diffusion energy-based model for interpretable text modeling. In *International Conference on Machine Learning*.

Lin Zheng, Jianbo Yuan, Lei Yu, and Lingpeng Kong. 2023. A reparameterized discrete diffusion model for text generation.