Effective Integration of Text Diffusion and Pre-Trained Language Models with Linguistic Easy-First Schedule

Yimin Ou[1], **Ping Jian**[1,2*]
[1]School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China
[2]Beijing Engineering Research Center of High Volume Language Information Processing
and Cloud Computing Applications, Beijing Institute of Technology, Beijing, China
{ymou, pjian}@bit.edu.cn

Abstract

Diffusion models have become a powerful generative modeling paradigm, achieving great success in continuous data patterns. However, the discrete nature of text data results in compatibility issues between continuous diffusion models (CDMs) and pre-trained language models (PLMs). That is, the performance of diffusion models even degrades when combined with PLMs. To alleviate this issue, we propose to utilize a pre-trained decoder to convert the denoised embedding vectors into natural language instead of using the widely used rounding operation. In this way, CDMs can be more effectively combined with PLMs. Additionally, considering that existing noise schedules in text diffusion models do not take into account the linguistic differences among tokens, which violates the easy-first policy for text generation, we propose a linguistic easy-first schedule that incorporates the measure of word importance, conforming to easy-first-generation linguistic features and bringing about improved generation quality. Experiment results on the E2E dataset and five controllable tasks show that our approach can combine the merits of CDMs and PLMs, significantly outperforming other diffusion-based models.

Keywords: text diffusion models, pre-trained language models, linguistic easy-first schedule

1. Introduction

Diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020) have recently emerged as state-of-the-art generative models, achieving high-quality synthesis results in the realm of modeling continuous data such as images (Ho et al., 2020), audio (Kong et al., 2020), and video (Ho et al., 2022). Diffusion models have also achieved great success in controllable generation and text-to-image systems, such as DallE 2 (Ramesh et al., 2022) and Imagen (Saharia et al., 2022).

The success of diffusion models in controllable generation makes them attractive for the text domain. Prior works have explored two representative diffusion processes for text generation, i.e., discrete diffusion (Hoogeboom et al., 2021; Austin et al., 2021), and continuous diffusion (Li et al., 2022). Discrete diffusion extends diffusion models to discrete state spaces while continuous diffusion performs the diffusion process in continuous latent representations of word embeddings and decodes the continuous generations with a rounding step.

Although there has been great progress, extending diffusion models to generate text data is still a challenging task. We observe that when combined with the pre-trained language models (PLMs) (Devlin et al., 2019; Lewis et al., 2020), the dimensionality of word embeddings becomes significantly high (e.g., 768 for BERT). In such cases, the KNN rounding operation fails to effectively decode the high-dimensional embedding vectors into natural language, resulting in incompatibility between the continuous diffusion models (CDMs) and PLMs, i.e., the performance of CDMs even degrades when combined with PLMs (Li et al., 2022).

Besides, existing noise schedules in text diffusion models do not consider the linguistic differences among tokens in a sequence, which violates the easy-first policy for text generation, causing the inaccurate generation of keywords and rare words. Therefore, there is an urgent need for a more effective method of integrating continuous diffusion models and pre-trained language models while considering the word importance in the noise schedule. In this work, we propose Diffusion-LEF, which aims to achieve the aforementioned goals. Specifically, we utilize a pre-trained encoder BERT (Devlin et al., 2019) to transform discrete text into embedding vectors, which are then subjected to diffusion operations. On the decoding side, we rely on a pre-trained BART Decoder (Lewis et al., 2020) to directly convert the denoised embedding vectors into natural language text, without the need for rounding operations. In addition, we introduce a linguistic easy-first schedule that considers linguistic features as the metric. This schedule aligns with the principle of generating text in an easy-first manner and ultimately enhances the quality of text generation.

Plug-and-play controllable generation (Dathathri et al., 2019) enables the generation of text with controllable properties (such as sentiment, style, or topic) without the need for further fine-tuning or retraining of the language

[*] Corresponding author

model. And continuous diffusion models have been argued to be effective in plug-and-play controllable generation tasks (Li et al., 2022). Hence, we follow the previous work and conduct experiments on the E2E dataset Novikova et al. (2017) and five controllable tasks including Semantic Content, Parts-of-Speech, Syntax Tree, Syntax Spans, and Length. The results indicate our Diffusion-LEF achieves competitive performance compared with recent baseline models with respect to both generation quality and fine-grained control ability.

To sum up, the main contributions of this work are as follows:

- We propose Diffusion-LEF, which effectively combines with the pre-trained language model BERT, leveraging the merits of both the diffusion models and pre-trained language models.
- We introduce a linguistic easy-first schedule that takes into account the linguistic features of words, bringing about higher-quality text generation.
- Experiments show that Diffusion-LEF achieves competitive performance with recent baseline models on different controllable generation tasks.

2. Related Works

2.1. PLMs for Text Generation

Recently, pre-trained language models (PLMs) have achieved significant success in text generation tasks (Qian et al., 2021). Most PLMs adopt an auto-regressive paradigm to generate text during pre-training and fine-tuning (Li et al., 2021). For example, the work based on GPT (Radford et al., 2019; Brown et al., 2020) converts different tasks into language modeling by predicting tokens sequentially. BART (Lewis et al., 2020) utilizes an auto-regressive decoder to recover corrupted text during pre-training. T5 (Raffel et al., 2020) masks spans of words in the input text, and then predicts the masked tokens in a sequential manner.

Since PLMs have achieved remarkable performance on various text generation tasks, we expect to integrate PLMs into text diffusion models to improve the quality of generated text. He et al. (2022) have explored the combination of pre-trained denoising language models with absorbing-state discrete diffusion models. This integration allows for leveraging the strengths of both models, resulting in a more comprehensive approach. However, their method is only effective for discrete diffusion models, and when combined with continuous diffusion models, it actually leads to a performance decline. In this work, we introduce a more effective way of combining continuous diffusion models with pre-trained language models, thus combining their respective advantages.

2.2. Diffusion Models for Text Generation

The great success of diffusion models in continuous domain (Ho et al., 2020) has attracted researchers to explore modeling in discrete domains, e.g., text generation. Existing text diffusion models can be broadly divided into two categories: discrete diffusion (Hoogeboom et al., 2021; Austin et al., 2021), and continuous diffusion (Li et al., 2022). Discrete diffusion performs the diffusion process on discrete text tokens, while continuous diffusion is conducted on continuous signals.

Sohl-Dickstein et al. (2015) first introduced the diffusion process in the discrete domain and proposed a binomial diffusion process to predict the binary representation of continuous data. Recent works have further explored diffusion processes that are more applicable to text. Hoogeboom et al. (2021) respectively explores the diffusion process for discrete states with categorical transition kernels, uniform transition kernels, and absorbing kernels. However, replacing continuous diffusion with a discrete corruption process affords some flexibility (Dieleman et al., 2022).

There are also some works exploring continuous diffusion for textual data modeling. Bit Diffusion (Chen et al., 2022) encodes discrete data as binary bits, treating these bits as real-valued features. DiffusionLM (Li et al., 2022) applies standard diffusion operations on the word embedding space and uses the rounding technique to map continuous space to discrete space during the reverse process. DiffuSeq (Gong et al., 2023) extends DiffusionLM to sequence-to-sequence settings by using an encoder-only Transformer and partial noising to define the diffusion process. Different from DiffuSeq, SeqDiffuSeq (Yuan et al., 2023) uses an encoder-decoder Transformer architecture and proposes adaptive noise schedule techniques. Despite making some progress, previous work ignored the differences between tokens in a sequence, leaving significant room for improvement in terms of text generation quality. In this work, we aim to utilize a linguistic easy-first schedule to improve it.

2.3. Plug-and-Play Controllable Generation

Plug-and-play controllable generation aims to produce text with controllable attributes, allowing users to manipulate specific attributes without any further fine-tuning or re-training. Most plug-and-play approaches adopt an auto-regressive paradigm to generate texts: Dathathri et al. (2019)

utilized gradient-based methods to modify the hidden representations of an auto-regressive language model to conform to specific control guidance. Yang and Klein (2021) introduced a technique that involves adjusting the significance of predicted tokens using reweighting methods. Additionally, Krause et al. (2021) and Liu et al. (2021) expanded on this approach by fine-tuning a smaller language model to enhance the weighting of token predictions.

The work closest to ours is PPLM (Dathathri et al., 2019), which has been successful on attribute control (e.g., topic), but less effective on complex control tasks (e.g., syntactic structure). In this work, we achieved plug-and-play controllable generation by updating the gradient of the middle latent variables with classifier guidance during the diffusion process and obtained promising results in multiple complex control tasks.

3. Preliminary

Diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020) are a class of latent variable generative models, that realize the generation of target data from noise (sampled from a simple distribution). It usually contains two processes: the forward process and the reverse process.

3.1. Forward Process

The forward process gradually disrupts the data sample x_0 using random noise. Specifically, given an input data sample $x_0 \sim q(x)$, the input data is perturbed by gradually adding a small amount of Gaussian noise until the input data becomes completely noisy. The forward process produces a Markov chain consisting of the hidden variables $x_1, \cdots, x_T$:

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

where $\beta_t \in (0,1)$ is the pre-defined scaling ratio of the noise variance at step t. Following a pre-defined noise schedule, β_t increases as the timestep grows, eventually corrupting x_0 into a random noise x_T. Then, based on the reparameterization trick, any intermediate latent variable x_t can be sampled from x_0 in closed form:

$$q\left(x_t \mid x_0\right)=\mathcal{N}\left(x_t ; \sqrt{\bar{\alpha}_t} x_0, \sqrt{1-\bar{\alpha}_t}\ \boldsymbol{I}\right), \quad (2)$$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$.

3.2. Reverse Process

The reverse process aims to learn the inverse process $p(x_{t-1}|x_t)$ of the forward process, which enables step-by-step denoising to recover the desired data sample x_0 from the random noise x_T. Since the true $p(x_{t-1}|x_t)$ in the reverse process is intractable, a neural network $p_\theta\left(x_{t-1} \mid x_t\right)$ is defined to approximate this distribution. When β_t is small enough, $p_\theta\left(x_{t-1} \mid x_t\right)$ can be modeled as a Gaussian distribution with two parameters: mean $\mu_\theta\left(x_t, t\right)$ and variance $\Sigma_\theta\left(x_t, t\right)$. Therefore, $p_\theta\left(x_{t-1} \mid x_t\right)$ can be defined as follows:

$$p_\theta\left(x_{t-1} \mid x_t\right)=\mathcal{N}\left(x_{t-1} ; \mu_\theta\left(x_t, t\right), \Sigma_\theta\left(x_t, t\right)\right), \quad (3)$$

where $\mu_\theta(\cdot)$ and $\Sigma_\theta(\cdot)$ are parameterized by a denoising network x_θ like U-Net (Ronneberger et al., 2015) or Transformer (Vaswani et al., 2017).

The learning objective of diffusion models is trained to maximize the marginal likelihood of $\log p_\theta\left(x_0\right)$ by minimizing the variational lower bound (Sohl-Dickstein et al., 2015):

$$\begin{aligned}
\mathcal{L}_{\text{vlb}} &= \mathbb{E}_q\left[D_{\text{KL}}\left(q\left(x_T \mid x_0\right) \| p_\theta\left(x_T\right)\right)\right] \\
&+\mathbb{E}_q\left[\sum_{t=2}^T D_{\text{KL}}\left(q\left(x_{t-1} \mid x_t, x_0\right) \| p_\theta\left(x_{t-1} \mid x_t, t\right)\right)\right] \\
&-\log p_\theta\left(x_0 \mid x_1\right),
\end{aligned} \quad (4)$$

where $\mathbb{E}_q(\cdot)$ denotes the expectation over the joint distribution $q(x_{0:T})$. However, this objective is usually unstable and requires many optimization tricks to stabilize (Nichol and Dhariwal, 2021). To address this issue, we follow Ho et al. (2020) to extend and reweight each KL divergence term in L_{vlb} and obtain a mean squared error loss:

$$\mathcal{L}_{\text{simple}}=\sum_{t=1}^T \mathbb{E}_q\left[\left\|\mu_t\left(x_t, x_0\right)-\mu_\theta\left(x_t, t\right)\right\|^2\right], \quad (5)$$

where $\mu_t(\cdot)$ is the mean of the posterior $q\left(x_{t-1} \mid x_t, x_0\right)$, and $\mu_\theta(\cdot)$ is the predicted mean of $p_\theta\left(x_{t-1} \mid x_t\right)$, which is predicted by the parameterized neural models. Through different parameterization strategies, the prediction objective can also be either the noise (Ho et al., 2020) or original data x_0 (Li et al., 2022).

4. Approach

In this section, we present the main design of our proposed Diffusion-LEF for controllable text generation. The overall diagram of Diffusion-LEF is shown in Figure 1.

4.1. Diffusion with pre-trained language models

To perform standard diffusion operations on discrete text, we utilize a learnable embedding layer or an encoder $E()$ to bridge the discrete space and continuous space. In order to leverage the prior knowledge of the language model, we can directly replace the embedding layer with the pre-trained

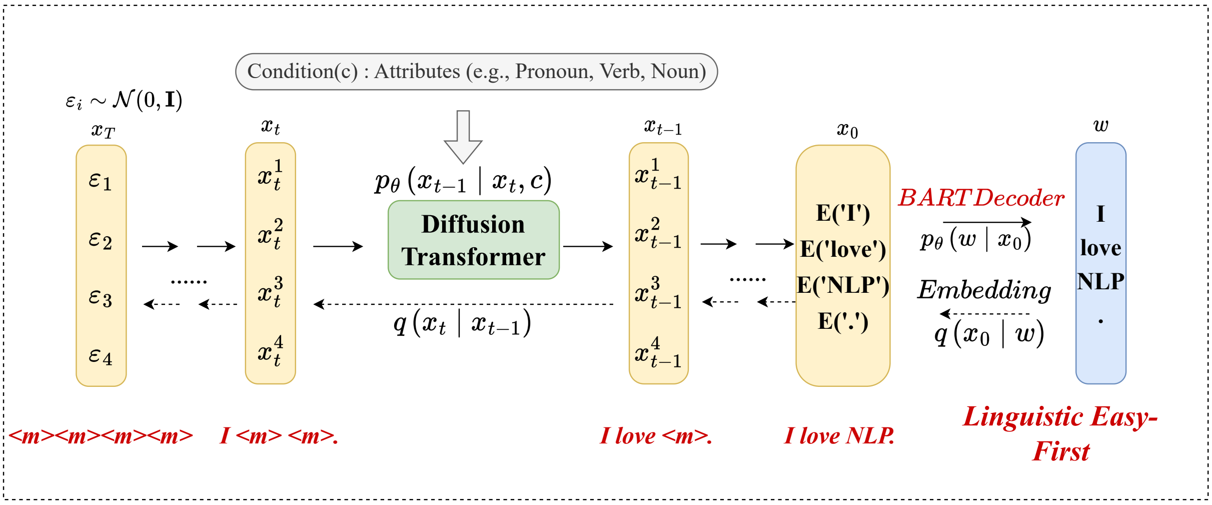

Figure 1: The overview of Diffusion-LEF.

language model. Specifically, given the input sentence d with l tokens $d = w_{1:l}$, the encoder $E()$ maps the discrete tokens to the continuous space and gets the hidden representations of input tokens as the initial state in diffusion models:

$$x_0 = w_{1:l} = E(w_{1:l}), \tag{6}$$

Later, we perform the standard diffusion operation on the tokens' latent variable representation. For decoding, we use a pre-trained BART Decoder $D()$ to reconstruct the original input:

$$w \approx \tilde{w} = D(x) = D(E(w)), \tag{7}$$

BART is a pre-trained language model that follows an encoder-decoder architecture. It is specifically trained as a denoising autoencoder, where it learns to reconstruct uncorrupted language text given input language utterances with masked tokens. This training process enables BART to effectively generate clean and coherent text output. By contrast, T5 is pre-trained to generate masked tokens of given corrupted text rather than fluent text. Therefore, BART is more suitable for our task, and we use BART-base as the decoder. To enhance efficiency, we freeze the parameters of BART while retaining only the trainable parameters of the denoising network $x_\theta()$.

4.2. Linguistic Easy-First Schedule

Existing noise schedules in text diffusion models are mainly derived from image generation tasks, such as the linear schedule (Ho et al., 2020) and the cosine schedule (Dhariwal and Nichol, 2021). These noise schedules do not consider the linguistic differences, i.e., importance and frequency, among tokens in a sequence. They treat all tokens equally in both forward and reverse process, which violates the easy-first policy (Kasai et al., 2020) for non-autoregressive text generation. The easy-first policy dictates that the model tends to first generate commonly used words as context for generating rare words later on. Failure to follow this policy can lead to tricky problems, such as inaccurate generation of key or rare words.

We propose the linguistic easy-first schedule to apply the easy-first policy to diffusion text generation. First, we need to define the importance I of words in a sentence. In this work, the importance I is defined based on their relevance and the amount of information they convey:

Word Relevancy

We use the TextRank (Mihalcea and Tarau, 2004) score as the metric to assess word relevancy within a sentence. By calculating the score of each node based on the weight between nodes, we determine the importance of each word in the sentence. A higher score assigned to a node indicates greater significance of the corresponding word within the sentence.

In a sentence, if a word w_i corresponds to a node v_i, and there exists an edge between node v_i and another node v_j corresponding to another word w_j, then the weight of the edge is defined as follows:

$$weight(v_i, v_j) = \frac{1}{\left|Out_{(v_j)}\right|}, \tag{8}$$

where $Out_{(v_j)}$ represents the set of outdegree of node v_j. The score of node v_i is defined as follows:

$$\begin{aligned} & score(v_i) = (1-d) \\ & +d \textstyle\sum_{v_j \in In(v_i)} \frac{weight(v_j, v_i)}{\sum_{v_k \in Out(v_j)} weight(v_j, v_k)} score(v_j), \end{aligned} \tag{9}$$

where $In_{(v_i)}$ represents the set of indegree of node v_i, d represents the damping coefficient, typically set to 0.85.

The amount of Information

We use entropy H (Bentz and Alikaniotis, 2016; He et al., 2022) to measure the amount of information of word *w* within a sentence. A word with higher entropy might contain greater unpredictability and information content in the given context and thus is more important compared to words with lower entropy. The formula for calculating entropy is:

$$H(w) = -p(w)\,\mathsf{log}(p(w)), \tag{10}$$

$$p(w) = \frac{f_w}{\sum_{j=1}^{V} f_j}, \tag{11}$$

where $p(w)$ represents the probability of the word w and f is the word frequency in the corpus.

In practice, we combine two metrics of word relevancy and the amount of information(normalized) to determine the importance I of the word w in one sentence d as follows:

$$I(w) = \frac{score(w)}{\sum_{w' \in d} score\left(w'\right)} + \frac{H(w)}{\sum_{w' \in d} H\left(w'\right)}, \tag{12}$$

Based on the introduced importance I of words in a sentence, we sort the words in descending order according to their importance and divide them into m buckets $W_{1:m}$, where the lower-indexed buckets contain more important words. During the forward process, we add noise to words with higher importance before words with lower importance, so that during the reverse process, easy (low importance) words emerge earlier than hard (high importance) words, which conforms to the easy-first-generation linguistic features and helps to achieve better generation quality. Specifically, at each step t, we add a small amount of Gaussian noise to the hidden representation of word w_i in bucket $W_{\lvert \frac{tm}{T} \rvert}$:

$$q\left(w_{i,t+1} \mid w_{i,t}\right) = \mathcal{N}\left(w_{i,t+1}; \sqrt{(1-\beta_t)}w_{i,t}, \beta_t I\right), \tag{13}$$

where the hyperparameter β_t is the amount of noise added at diffusion step t.

Li et al. (2022) observes that the nearest neighbors of words in the embedding space stay constant after corruption and attributes this phenomenon to the small initial noise scale in traditional schedules. Thus, it introduces the sqrt schedule which has a higher initial noise scale and increasing rate, while gradually slowing down to avoid producing too many highly corrupted latent variables. Following the work of Li et al. (2022), we apply the sqrt noise schedule to gradually increase β_t.

$$\beta_t = 1 - \sqrt{t/T + s}, \tag{14}$$

where s is a small constant that corresponds to the starting noise level.

We incorporate the measure of word importance into the noise schedule and name it as *linguistic easy-first schedule*, which fully considers the linguistic features of tokens. During the forward noising process, harder (high importance) words be added with less noise, which is beneficial for maintaining training stability.

4.3. Self-Conditioning

In the reverse process of standard diffusion models, the denoising network only makes predictions based on the current latent variable x_t and time step t. Chen et al. (2022) proposed the self-conditioning technique, which adds the predicted output from the previous timestep to the denoising network. This is formulated as $\widetilde{x}_t = x_\theta\left(x_t,\ t, \widetilde{x}_{t+1}\ \right)$. It has demonstrated significant improvements in text generation quality and has become a widely-used technique in text diffusion models (Dieleman et al., 2022; Gao et al., 2022).

We apply the self-conditioning technique to our Diffusion-LEF. During the inference state, the sampling procedure is essentially iterative. Therefore, the sampling process does not require any additional modifications. However, we need to modify the training procedure due to the unavailability of $\widetilde{x}_{t+1}$. Specifically, for each training step t, with probability $p = 0.5$, we do not provide any estimate of the data for self-conditioning. In this case, $\widetilde{x}_{t,\emptyset} = x_\theta\left(x_t,\ t,\ \emptyset\ \right)$ is trained by setting the previous predictions $\widetilde{x}_{t+1}$ to 0. With probability $1-p$, we mimic the inference behavior by first computing the value of $\widetilde{x}_{t,\emptyset} = x_\theta\left(x_t,\ t,\ \emptyset\ \right)$ and then computing the additional estimate $\widetilde{x}_t = x_\theta\left(x_t,\ t,\ sg(\widetilde{x}_{t,\emptyset})\ \right)$, where $sg()$ is the stopping gradient operation. In the second case, we do not backpropagate through the first estimated $\widetilde{x}_{t,\emptyset}$.

4.4. Denoising Network Architecture

In the forward process, the initial latent variable x_0 is gradually noised into a series of noisy latent variables $x_{1:T}$. The reverse process needs to gradually denoise x_T back to x_0. We define the denoising network as $x_\theta(x_t,\ t)$, which is based on the Transformer (Vaswani et al., 2017) architecture with 12 layers and a hidden dimension of 768.

To obtain the reconstruction, we project the latent x_t onto the input dimension of the Transformer, pass it through the Transformer, and subsequently process it with a LayerNorm (Ba et al., 2016) and a linear layer.

For the decoding strategy, we follow DiffusionLM (Li et al., 2022) to use the Minimum Bayes Risk (MBR) decoding (Kumar and Byrne, 2004). When applying MBR decoding, we make use of the generated sequence candidates set C for each sample. By calculating the expected risk R for each candidate sequence within the set, MBR decoding aims to find the candidate sequence $s*$ that minimizes this expected risk.

$$s^* = \arg\min_{s \in \mathcal{C}} R(s) = \arg\min_{s \in \mathcal{C}} \frac{1}{|\mathcal{C}|} \sum_{s' \in \mathcal{C}} r\left(s, s'\right), \tag{15}$$

where $r(\cdot, \cdot)$ represents a specific risk function, and we use the negative BLEU score following DiffusionLM. The sequence candidates within the candidate set C are generated from the diffusion models using various random seeds.

4.5. Controllable Text Generation with Diffusion-LEF

Our approach follows the setting of Plug-and-Play controllable generation as DiffusionLM (Li et al., 2022), which utilizes an external classifier to perform control over the latent variables x_t in each intermediate step $t \in [0, T]$ of the diffusion process:

$$p\left(x_{0:T} \mid c\right) = \prod_{t=1}^{T} p\left(x_{t-1} \mid x_t, c\right), \tag{16}$$

Following the conditional independence assumption in previous work on controlling diffusion (Yang and Klein, 2021), we decompose the joint inference problem of $p(x_{0:T}|c)$ into a series of control problems at each diffusion step t:

$$\begin{aligned} p\left(x_{t-1} \mid x_t, c\right) &\propto p\left(x_{t-1} \mid x_t\right) \cdot p\left(c \mid x_{t-1}, x_t\right) \\ &= p\left(x_{t-1} \mid x_t\right) \cdot p\left(c \mid x_{t-1}\right), \end{aligned} \tag{17}$$

Therefore, for the t^{th} step, we run gradient updates on x_t to generate x_{t-1}:

$$\begin{aligned} \nabla_{x_{t-1}} \log p\left(x_{t-1} \mid x_t, c\right) &= \lambda \nabla_{x_{t-1}} \log p\left(x_{t-1} \mid x_t\right) \\ &+ \nabla_{x_{t-1}} \log p\left(c \mid x_{t-1}\right), \end{aligned} \tag{18}$$

where log $p(x_{t-1}|x_t)$is parameterized by the Diffusion Transformer and $logp(c|x_{t-1})$ is parameterized by a neural network classifier. Both terms are differentiable. Additionally, λ is a fluency regularization hyperparameter that trades off fluency and control to enhance generation quality.

5. Experiments

5.1. Tasks and Datasets

We trained Diffusion-LEF on the E2E datasets (Novikova et al., 2017) which is composed of 50,000 restaurant reviews that have been labeled according to 8 different fields. Then we apply our controllable generation method to four classifier-guided control tasks, i.e., Semantic Content, Parts-of-Speech, Syntax Tree, Syntax Spans, and one classifier-free control task, i.e., Length.

For every control task, we sample 200 control targets c from the validation splits, and for each control target, we generate 50 corresponding samples. To evaluate the fluency of text generated by Diffusion-LEF, we use a teacher LM (i.e., a carefully fine-tuned GPT-2 model) and report the perplexity of the generated text under the teacher LM. Lower perplexity indicates better sample quality and fluency. For each control task, we define accuracy metrics as follows:

Semantic Content. For a given field (e.g., *eatType*) and value (e.g., *coffee shop*), the task is to generate a sentence that includes the format of field = value. We evaluate the accuracy of the generated sentences by examining the exact match rate of the word mentions for 'value'.

Parts-of-Speech. For a given sequence of parts-of-speech (POS) tags (e.g., *Det Noun Verb Det Noun*), the task is to generate a sentence with the same length and follow the exact given POS tag sequence (e.g., *The cat chased the mouse*). We evaluate the accuracy by checking for exact matches between the word-level POS tags and the corresponding tags generated by an oracle POS tagger.

Syntax Tree. For a given syntactic parse tree, the task is to generate a sentence with the same parse tree. We evaluate the accuracy by first parsing the generated sentence with an off-the-shelf parser and report the F1 scores compared to the given parse.

Syntax Spans. For a given (span, syntactic category) pair (e.g., *(1, 3, NP)*), the parse tree of the generated sentence should match the given syntactic category over the given spans. We evaluate the accuracy of the sentence by the exact match rate of the given spans.

Length. For a given target length (e.g., *30*), the task is to generate a sentence within ±2 of the given target. We evaluate the accuracy by the match rate of the sentence lengths.

5.2. Baselines

In our controllable text generation experiments, we conducted a comparative analysis between Diffusion-LEF and the following state-of-the-art baseline models.

PPLM (Dathathri et al., 2019) increases the classifier probabilities and language model probabilities by running gradient ascent on the pre-trained language model activations. Since the classifier of PPLM lacks location information, we only apply PPLM to the task of controlling semantic content.

FUDGE (Yang and Klein, 2021) reweights the tokens predicted by a pre-trained language model. It uses a discriminator that takes a sequence of prefixes and predicts whether the full sequence satisfies the constraints.

Methods	Semantic Content		Part-of-Speech		Syntax Tree		Syntax Spans		Length	
	Accuracy↑	Fluency↓	Accuracy↑	Fluency↓	Accuracy↑	Fluency↓	Accuracy↑	Fluency↓	Accuracy↑	Fluency↓
PPLM	9.9	5.32	-	-	-	-	-	-	-	-
FUDGE	69.9	2.83	27.0	7.96	17.9	3.39	54.2	4.03	46.9	3.11
DiffusionLM	81.2	2.55	90.0	5.16	86.0	3.71	93.8	2.53	99.9	2.16
DiffusionLM+BERT	77.4	2.68	86.2	5.43	82.3	3.92	89.3	3.13	99.9	2.68
Diffusion-LEF	81.7	2.46	91.2	5.09	86.3	3.68	94.4	2.48	99.9	2.14
Diffusion-LEF+BERT	**82.4**	**2.32**	**92.4**	**4.82**	**89.4**	**3.48**	**95.5**	**2.36**	**100**	**2.10**

Table 1: Main results on five controllable generation tasks.

Methods	Semantic Content	Part-of-speech	Syntax Tree	Syntax Spans	Length
DiffusionLM	3.56	3.63	3.61	3.42	3.81
DiffusionLM+BERT	2.81	3.10	2.96	3.04	3.20
Diffusion-LEF	3.89	4.05	4.12	3.72	3.98
Diffusion-LEF+BERT	**4.32**	**4.54**	**4.61**	**4.21**	**4.14**

Table 2: Human evaluation scores of different methods on five controllable generation tasks.

	Semantic Content	
	Accuracy	Fluency
Diffusion-LEF	81.7	2.46
w/o Ling. Sche.	81.5	2.52
w/o BART Decoder	81.4	2.49

Table 3: Ablation studies on the Semantic Content task.

DiffusionLM (Li et al., 2022) learns an embedding to map the discrete text into the continuous space where it performs the Gaussian diffusion process. Additionally, it introduces a rounding step designed to map the embeddings back into discrete texts.

5.3. Implementation Details

Our Diffusion-LEF is based on Transformer architecture with 12 layers and a hidden dimension of 768, with a sequence length $n = 64$, diffusion steps $T = 500$, and a sqrt noise schedule. We set the embedding dimension to 128. When combined with the PLM BERT, the embedding dimension is set to 768. In this work, we use the BERT-base with about 110M parameters and freeze the parameters in BERT. The number of word buckets *m* is set to 3. We learn Diffusion-LEF with the AdamW optimizer (Loshchilov and Hutter, 2017) for 20,000 steps with learning rate of 3e-4, dropout probability of 0.1, and batch size of 64. We use a linear warmup schedule starting with 1,000 warmup steps. All experiments are conducted on 4 NVIDIA RTX A6000 GPUs.

5.4. Experimental Results

Main Results We report the main experimental results of our Diffusion-LEF and baselines on five controllable text generation tasks in Table 1. Diffusion-LEF achieves the highest Fluency and Accuracy scores on five controllable generation tasks, indicating that Diffusion-LEF has excellent text generation quality and fine-grained control ability when equipped with the linguistic easy-first schedule. Compared with the non-diffusion methods PPLM and FUDGE, the diffusion-based methods DiffusionLM and Diffusion-LEF both achieved great improvements (e.g., 90.0 for DiffusionLM and 91.2 for Diffusion-LEF vs. 27.0 for FUDGE on the Part-of-speech task), indicating the applicability of diffusion models on controllable generation tasks.

It is observed that when combined with the pretrained language model BERT, the performance of DiffusionLM has declined. One explanation for this phenomenon is that the rounding operation to bridge continuous space and discrete space suffers from significantly high dimensions, which results in incompatibility between the continuous diffusion models and the PLMs. Instead of employing rounding operations, our Diffusion-LEF relies on the BART decoder to convert the denoised embedding vectors into natural language. This strategy effectively resolves the challenge of high di-

Time Step *T-t*	Sentences
Input (Semantic Content)	**food : Chinese**
0	[mask] [mask] [mask] [mask] [mask] [mask] [mask] [mask]
200	The [mask] restaurant serves [mask] [mask] cuisine.
400	The [mask] restaurant serves authentic [mask] cuisine.
500	The Chinese restaurant serves authentic Sichuan cuisine.
Input (Length)	**10**
0	[mask] [mask] [mask] [mask] [mask] [mask] [mask] [mask] [mask] [mask]
200	The restaurant is [mask]and has a [mask] [mask].
400	The restaurant is [mask] and has a nice ambiance.
500	The restaurant is cozy and has a nice ambiance.

Table 4: Examples of the intermediate generated text of Diffusion-LEF on the Semantic Content and Length task.

mensionality. As demonstrated in Table 1, when combined with the PLM BERT, our Diffusion-LEF exhibits improved performance compared to DiffusionLM. This indicates that our method can efficiently integrate with the PLMs and thus make full use of the merits of both.

Human Evaluation In addition to automatic metrics, human evaluation is also highly valuable for text generation tasks. To better demonstrate the performance of Diffusion-LEF, we invite five graduate students with proficiency in English as human annotators to evaluate the text generated by different models. The details are as follows: For each control task, we randomly select 30 samples from four models: DiffusionLM, DiffusionLM+BERT, Diffusion-LEF, and Diffusion-LEF+BERT. Five annotators are asked to rate the samples using five scores [1, 2, 3, 4, 5], where higher scores represent higher quality. The scoring criteria consist of two factors: (i) fluency, which evaluates the readability and fluency of the given sentence, and (ii) controllability, which evaluates whether the given sentence aligns with the specified control condition. To ensure fairness, the human evaluation is carried out in a blind manner, where the annotators are kept unaware of which model the output sequence is related to. Table 2 shows the results of the human evaluation. We can observe that DiffusionLM+BERT has the lowest rating score, while Diffusion-LEF+BERT has the highest rating score, which shows a similar trend with the automatic metrics. That is, our proposed Diffusion-LEF and Diffusion-LEF+BERT can achieve better results than DiffusionLM, which fully proves the superiority of our proposed method.

5.5. Ablation Study

Our Diffusion-LEF includes several key designs, i.e., linguistic easy-first schedule and the usage of BART Decoder. Here, we conduct the ablation studies on the Semantic Content task to verify their effectiveness.

In Table 3, Ling. Sche. is short for linguistic easy-first schedule, w/o Ling. Sche. represents the removal of the linguistic easy-first schedule from Diffusion-LEF and instead using the fixed sqrt schedule, w/o Ling. Sche represents the removal of BART Decoder and instead uses the rounding operation. We can observe that after removing the corresponding component, the performance of Diffusion-LEF drops consistently, indicating the effectiveness of the designs we employed.

5.6. Case Study

In order to visually show the generation process of Diffusion-LEF, we select two examples for presentation, as shown in Table 4. It can be observed that on the Semantic Content task, easy (low importance) words such as ”the”, ”restaurant” and ”cuisine” are generated earlier, while hard (high importance) words such as ”Chinese” and ”Sichuan” are generated with the increase of diffusion steps. Additionally, the generated sentence also includes the format of ”field = value” (food = Chinese). On the Length task, the words ”the”, ”is” and ”a” are also generated before ”cozy” and ”nice”. The generated sentence also satisfies the required length criteria. This fully shows that the generation process of Diffusion-LEF follows the easy-first-generation linguistic features, while also enabling controlled generation.

6. Conclusion

In this work, we propose Diffusion-LEF, a text Diffusion model effectively combined with pre-trained language model BERT, which leverages the merits of both diffusion models and pre-trained language models. In addition, we propose the linguistic easy-first schedule that incorporates the measure of word importance, conforming to easy-first-generation linguistic features and bringing about improved generation quality. Through experiments conducted on the E2E dataset and five controllable tasks including Semantic Content, Parts-of-Speech, Syntax Tree, Syntax Spans, and Length, we demonstrate the superior performance of Diffusion-LEF compared with recent baseline models in terms of generation quality and fine-grained control ability.

Limitations

This work aims to efficiently integrate diffusion and pre-trained language models for controllable text generation with the linguistic easy-first schedule. It is worth noting that the combination of pre-trained language models may introduce biases learned from the pre-training corpus into the generated texts. Another important limitation is the slow speed of sample generation from diffusion models due to the iterative nature of the sampling process. Although some existing works, such as DDIM (Song et al., 2020) and Dpm-solver (Lu et al., 2022), have been dedicated to improving the inference speed, they may cause the mismatch of diffusion trajectories between training and inference. How to alleviate this matching problem will be a future research direction.

Acknowledgement

The authors would like to thank the organizers of LREC-COLING2024 and the reviewers for their helpful suggestions. This work is supported by the Open Projects Program of State Key Laboratory of Multimodal Artificial Intelligence Systems (No. MAIS-2023-13) and the National Natural Science Foundation of China (No. 62376130).

Bibliographical References

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

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Christian Bentz and Dimitrios Alikaniotis. 2016. The word entropy of natural languages. *arXiv preprint arXiv:1606.06996*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, J. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. Henighan, Rewon Child, A. Ramesh, Daniel M. Ziegler, Jeff 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.

Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. 2022. Analog bits: Generating discrete data using diffusion models with self-conditioning. *arXiv preprint arXiv:2208.04202*.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, J. Yosinski, and Rosanne Liu. 2019. Plug and play language models: A simple approach to controlled text generation.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794.

Sander Dieleman, Laurent Sartran, Arman Roshannai, Nikolay Savinov, Yaroslav Ganin, Pierre H Richemond, Arnaud Doucet, Robin Strudel, Chris Dyer, Conor Durkan, et al. 2022. Continuous diffusion for categorical data. *arXiv preprint arXiv:2211.15089*.

Z Gao, J Guo, X Tan, Y Zhu, F Zhang, J Bian, and L Difformer Xu. 2022. 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. 2023. Diffuseq: Sequence to sequence text generation with diffusion models. In *The Eleventh International Conference on Learning Representations*.

Zhengfu He, Tianxiang Sun, Kuan Wang, Xuanjing Huang, and Xipeng Qiu. 2022. Diffusionbert: Improving generative masked language models with diffusion models.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. 2022. Video diffusion models.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. 2021. Argmax flows and multinomial diffusion: Learning categorical distributions. *Advances in Neural Information Processing Systems*, 34:12454–12465.

Jungo Kasai, James Cross, Marjan Ghazvininejad, and Jiatao Gu. 2020. Non-autoregressive machine translation with disentangled context transformer.

Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. 2020. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761*.

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. 2021. Gedi: Generative discriminator guided sequence generation.

Shankar Kumar and W. Byrne. 2004. Minimum bayes-risk decoding for statistical machine translation.

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.

Junyi Li, Tianyi Tang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. Pretrained language models for text generation: A survey. *arXiv preprint arXiv:2105.10311*.

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. 2022. Diffusion-lm improves controllable text generation. *Advances in Neural Information Processing Systems*, 35:4328–4343.

Alisa Liu, Maarten Sap, Ximing Lu, Swabha Swayamdipta, Chandra Bhagavatula, Noah A. Smith, and Yejin Choi. 2021. Dexperts: Decoding-time controlled text generation with experts and anti-experts.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

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. *arXiv preprint arXiv:2206.00927*.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into texts. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 404–411.

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

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017. The e2e dataset: New challenges for end-to-end generation.

Lihua Qian, Hao Zhou, Yu Bao, Mingxuan Wang, Lin Qiu, Weinan Zhang, Yong Yu, and Lei Li. 2021. Glancing transformer for non-autoregressive neural machine translation.

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

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. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. 2022. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR.

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

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

Kevin Yang and Dan Klein. 2021. Fudge: Controlled text generation with future discriminators. *arXiv preprint arXiv:2104.05218*.

Hongyi Yuan, Zheng Yuan, Chuanqi Tan, Fei Huang, and Songfang Huang. 2023. Seqdiffuseq: Text diffusion with encoder-decoder transformers.