Smoothie: Smoothing Diffusion on Token Embeddings for Text Generation

Alexander Shabalin HSE University Constructor University amshabalin@hse.ru

Viacheslav Meshchaninov
HSE University
Constructor University
vmeshchaninov@hse.ru

Dmitry Vetrov
Constructor University
dvetrov@constructor.university

Abstract

Diffusion models have achieved state-of-the-art performance in generating images, audio, and video, but their adaptation to text remains challenging due to its discrete nature. Prior approaches either apply Gaussian diffusion in continuous latent spaces, which inherits semantic structure but struggles with token decoding, or operate in categorical simplex space, which respect discreteness but disregard semantic relation between tokens. In this paper, we propose Smoothing Diffusion on Token Embeddings (SMOOTHIE), a novel diffusion method that combines the strengths of both approaches by progressively smoothing token embeddings based on semantic similarity. This technique enables gradual information removal while maintaining a natural decoding process. Experimental results on several sequence-to-sequence generation tasks demonstrate that SMOOTHIE outperforms existing diffusion-based models in generation quality. Furthermore, ablation studies show that our proposed diffusion space yields better performance than both the standard embedding space and the categorical simplex. Our code is available at https://github.com/ashabalin/smoothie.

1 Introduction

Diffusion models attracted a lot of attention in recent years as they show very high generation quality in image [41, 37], audio [12] and video [3] domains surpassing all previous approaches such as GANs [15] and Normalizing Flows [39]. Diffusion models work by introducing a forward process that gradually degrades an object by injecting Gaussian noise into it, and then learning the reverse process by denoising the object.

Applying diffusion models to text is challenging due to its discrete nature. Nevertheless, several works have explored ways to design suitable diffusion processes. One line of research proposes gradually removing information by replacing tokens with others sampled from a categorical distribution [2, 18, 31]. Another approach applies Gaussian diffusion to the latent space of token embeddings [27, 13]. Additionally, some studies leverage the discreteness of text by performing diffusion directly on the vocabulary probability simplex instead of the embedding space [25, 16].

Each of the described methods offers distinct advantages and limitations, as summarized in Table 1. Gaussian diffusion progressively removes semantic information: under the Euclidean semantic space hypothesis [17], the distinguishability of noisy tokens depends on their initial distances in the latent space. The addition of Gaussian noise gradually disrupts these distances, making the semantics of a latent representation increasingly difficult to recover. However, Gaussian diffusion does not account for the discrete nature of text, which complicates the mapping of generated latent vectors back to discrete tokens [27, 42].

On the other hand, categorical and simplex-based diffusion methods naturally preserve the discreteness of text and eliminate the need for an explicit decoding step. Nevertheless, they disregard

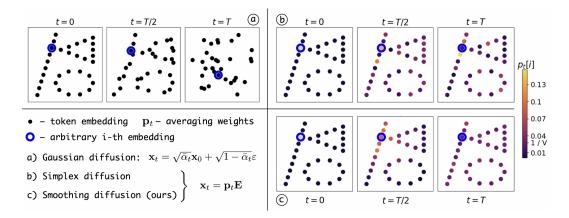


Figure 1: An illustration of the diffusion process for Gaussian, simplex, and smoothing diffusion methods. The key distinction between simplex and smoothing diffusion is that the latter incorporates semantic relationships between tokens during the noise addition process.

Table 1: Comparison of diffusion methods in terms of accounting for text discreteness and semantics.

	Categorical	Gaussian	Simplex	Smoothing (Ours)
Accounting for Discreteness	✓	X	✓	✓
Accounting for Semantics	×	✓	X	✓

semantic relationships between tokens during the noising process, resulting in a more erratic and less meaningful degradation of information.

In this paper, we propose SMOOTHIE, a smoothing diffusion framework that satisfies both properties. We represent each token with a vector based on distances between token embeddings. During the forward process, our diffusion mechanism gradually perturbs these distances, progressively dissolving semantic information. Like simplex diffusion, our method enables natural decoding from latent representations back to tokens. In theory, SMOOTHIE is applicable not only to text, but to any domain where data comes from a categorical distribution with inherent similarity between categories.

We evaluate SMOOTHIE on four sequence-to-sequence generation tasks and show that it outperforms existing diffusion-based approaches on the majority of tasks. Ablation studies further demonstrate that our method enables effective control over the trade-off between fluency and diversity of the generated text.

The main contributions of our work are as follows:

- 1. We propose a novel text diffusion framework that simultaneously respects the discrete nature of text and progressively removes semantic information from token representations during the forward process.
- 2. We show the practical effectiveness of our approach across multiple sequence-to-sequence generation tasks, providing empirical evidence for the advantages of our diffusion design.

2 Problem statement and background

Problem statement In this work, we focus primarily on sequence-to-sequence generation tasks, which can be formally defined as follows. Given a source sequence $\mathbf{w}^x = \{w_1^x, \dots, w_n^x\}$, the objective is to generate a corresponding target sequence $\mathbf{w}^y = \{w_1^y, \dots, w_m^y\}$. We consider parallel datasets, where each source sequence is paired with a known target sequence.

Gaussian diffusion model The diffusion process is defined in terms of a forward (noising) and a reverse (denoising) processes. Given an initial data point sampled from the data distribution, $\mathbf{x}_0 \sim p_{\text{data}}$, the forward process generates a sequence of progressively noisier latent variables $\mathbf{x}_1, \dots, \mathbf{x}_T$. Each

step in this sequence is defined by the transition $\mathbf{x}_t \sim q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}(\sqrt{\alpha_t}\mathbf{x}_{t-1}, \sqrt{1-\alpha_t}, \varepsilon)$, where the parameter $\alpha_t \in [0,1)$ controls the amount of noise injected at timestep t. This formulation also supports a direct sampling of \mathbf{x}_t from \mathbf{x}_0 using the marginal distribution $q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}(\sqrt{\bar{\alpha}_t}\mathbf{x}_0, \sqrt{1-\bar{\alpha}_t}, \varepsilon)$, where $\bar{\alpha}_t = \prod_{s=0}^t \alpha_s$ denotes the cumulative product of noise scales.

After the forward process is complete, a neural network f_{θ} is trained to reverse it by predicting the original data point \mathbf{x}_0 from the noisy input \mathbf{x}_t . During generation, the model iteratively denoises an initial sample $\mathbf{x}_T \sim \mathcal{N}(0, I)$, gradually reconstructing the data through the learned reverse process until it recovers \mathbf{x}_0 .

Embedding diffusion The most popular continuous text diffusion approaches create a latent space by mapping tokens to their embeddings [27, 13, 47]. Then the Gaussian diffusion process is used to corrupt a latent. The decoding is usually performed by mapping a generated embedding to the token corresponding to the closest embedding.

Simplex diffusion SSD-LM [16] and TESS [25] propose a simplex diffusion model. They map each token w to a k-logit simplex $\mathbf{s}^w \in \{\pm k\}^V$, where V is the size of the vocabulary and

$$\mathbf{s}_{(i)}^{w} = \begin{cases} +k, & i = w \\ -k, & \text{otherwise} \end{cases}$$
 (1)

Then the latent is represented as a sequence $\mathbf{S}_0 = (\mathbf{s}^{w_1^y}, \dots, \mathbf{s}^{w_m^y})$. Corruption is performed with the Gaussian diffusion process with noise variance multiplied by k^2 (k=5 by default), $\mathbf{S}_t = \sqrt{\bar{\alpha}_t}\mathbf{S}_0 + k\sqrt{1-\bar{\alpha}_t}\varepsilon$. The model input is calculated by first producing a probability simplex over vocabulary, $\mathbf{p}_t = \operatorname{softmax}(\mathbf{S}_t)$, and then averaging token embeddings with obtained weights, $\mathbf{p}_t\mathbf{E}$, where \mathbf{E} is a matrix of token embeddings.

3 Related work

Since the initial attempt to apply diffusion models to text generation [22], numerous studies have explored ways to better align the diffusion process with the specifics of textual data. D3PM [2] tried exploiting the semantic property of tokens by applying a discrete diffusion process that replaces tokens with semantically similar alternatives with higher probability. However, their experiments showed that simple token masking approach produces better empirical results.

Diffusion-LM [27] proposed applying Gaussian diffusion in the continuous latent space of token embeddings, while TEncDM [42] further demonstrated that context-dependent embeddings provide a more suitable latent space for continuous diffusion. Despite achieving strong generation quality, the downside of these methods is the requirement of an additional latent decoding step.

DiffuSeq-v2 [14] attempted to bridge the gap between discrete and continuous diffusion models by combining masking with Gaussian noise during the noising process. Another research direction [16, 25] focuses on mapping tokens to almost-one-hot simplex representations over the vocabulary and introducing Gaussian noise directly into this space. While this approach does not account for token semantics during noising, it preserves the discrete structure of text.

Our work is inspired by a different line of research developed in the image domain [40, 21], where semantic information is gradually removed by smoothing pixel values according to the heat dissipation principle. However, while being effective for continuous signals such as images, this strategy can not be directly applied to text due to its inherently discrete nature.

4 Smoothing diffusion

In this section, we introduce SMOOTHIE, a smoothing text diffusion model that incorporates both the discrete nature of text and the semantic relationships between tokens into the diffusion process. We will first derive the diffusion process for unconditional generation and then extend it to conditional generation. We provide an intuitive illustration of our approach, along with pseudo-code for the training and sampling procedures, in Fig. 1, Alg. 1, and Alg. 2, respectively.

4.1 Forward diffusion process

Let V denote the vocabulary size, and let $\mathbf{E} \in \mathbb{R}^{V \times d}$ be a fixed embedding matrix, where each row corresponds to a d-dimensional token embedding. To construct a latent space suitable for diffusion, we represent each token w_i^y in a target sequence \mathbf{w}^y with a vector of negative squared Euclidean distances between an embedding of token w_i^y and embeddings of all tokens in the vocabulary:

$$\mathbf{D}_0 = \mathbf{D}_0(\mathbf{E}_{\mathbf{w}^y}) = \left\{ -\frac{\|\mathbf{E}_{w_i^y} - \mathbf{E}_j\|^2}{2} \right\}_{i,j=1}^{m,V}$$
(2)

Here, $\mathbf{E}_{w_i^y}$ is the embedding of the i-th token in the sequence, and \mathbf{E}_j is the embedding of the j-th vocabulary token. To generate a trajectory of progressively noisier latents, we define a non-Markovian forward, or noising process:

Forward process
$$q(\mathbf{D}_{1:T}|\mathbf{D}_0) = \prod_{t=1}^T q(\mathbf{D}_t|\mathbf{D}_0) = \prod_{t=1}^T \mathcal{N}\left(\mathbf{D}_t \middle| \frac{1}{\sigma_t^2} \mathbf{D}_0, \delta^2 I\right)$$
(3)

The noise scheduler σ_t ($1 < \sigma_1 < \cdots < \sigma_T$) controls the amount of noise added at each timestep. The hyperparameter δ controls the stochasticity of the diffusion process and makes it non-deterministic. Following [40], we keep δ independent of the timestep t.

To construct the model input, we convert \mathbf{D}_t into a probability distribution over the vocabulary using the softmax function: $\mathbf{p}_t = \operatorname{softmax}(\mathbf{D}_t)$. In this formulation, each token is represented by the weights of Nadaraya-Watson kernel estimator applied over all embeddings in the vocabulary with Gaussian kernel whose bandwidth is defined by σ_t . The choice of a Gaussian kernel is motivated by the Euclidean semantic space hypothesis [17], which assumes that semantic similarity correlates with Euclidean proximity in embedding space. As a result, as σ_t increases, the probability mass—initially centered in a single token—gradually distributes between all other tokens, starting from the most semantically similar and ending with the most distant ones (see Fig. 1 (c)).

Note that our approach can be viewed as a generalization of a simplex-based diffusion [16, 25]. In particular, by replacing our Euclidean distance with trivial metric, we get the latent space formulation defined in Eq. 1, which ignores the semantic relationships between tokens. We prove this statement in Appendix C. In Section 5 we show that incorporating semantic similarity into the diffusion process is crucial for achieving better performance.

4.2 Reverse diffusion process

The reverse, or denoising process, starts with a sample from prior distribution $p(\mathbf{D}_T)$ and ends with the denoised data sample \mathbf{D}_0 . We define it as a Markov chain with Gaussian distributions:

Reverse process
$$p_{\theta}(\mathbf{D}_{0:T}) = p(\mathbf{D}_T) \prod_{t=1}^{T} p_{\theta}(\mathbf{D}_{t-1}|\mathbf{D}_t) = p(\mathbf{D}_T) \prod_{t=1}^{T} \mathcal{N}(\mathbf{D}_{t-1}|\mu_{\theta}(\mathbf{p}_t, t), \tilde{\delta}^2 I),$$
 (4)

where θ are trainable model parameters and $\tilde{\delta}^2$ is a noise variance used in the reverse process. Inspired by [40], we allow noise variance to change between the forward and reverse processes. That permits us to explicitly control the stochasticity of the generation trajectory, which significantly affects the model performance (see Section 5.1).

Our goal is to find such parameters θ , that minimize the marginal negative likelihood of data samples $p_{\theta}(\mathbf{D}_0) = \int p_{\theta}(\mathbf{D}_{0:T}) \mathrm{d}\mathbf{D}_{1:T}$. We optimize the negative log-likelihood by minimizing its variational upper bound:

$$-\log p_{\theta}(\mathbf{D}_0) = -\log \int \frac{p_{\theta}(\mathbf{D}_{0:T})q(\mathbf{D}_{1:T}|\mathbf{D}_0)}{q(\mathbf{D}_{1:T}|\mathbf{D}_0)} d\mathbf{D}_{1:T} \le -\mathbb{E}_q \log \frac{p_{\theta}(\mathbf{D}_{0:T})}{q(\mathbf{D}_{1:T}|\mathbf{D}_0)}$$
(5)

$$= -\mathbb{E}_q \left[\log \frac{p_{\theta}(\mathbf{D}_T)}{q(\mathbf{D}_T|\mathbf{D}_0)} + \sum_{t=2}^T \log \frac{p_{\theta}(\mathbf{D}_{t-1}|\mathbf{D}_t)}{q(\mathbf{D}_{t-1}|\mathbf{D}_0)} + \log p_{\theta}(\mathbf{D}_0|\mathbf{D}_1) \right]$$
(6)

$$= \mathbb{E}_{q} \left[\underbrace{\mathbf{D}_{\mathrm{KL}} \left[q(\mathbf{D}_{T} | \mathbf{D}_{0}) \| p(\mathbf{D}_{T}) \right]}_{L_{T}} + \sum_{t=2}^{T} \underbrace{\mathbf{D}_{\mathrm{KL}} \left[q(\mathbf{D}_{t-1} | \mathbf{D}_{0}) \| p_{\theta}(\mathbf{D}_{t-1} | \mathbf{D}_{t}) \right]}_{L_{t-1}} \underbrace{-\log p_{\theta}(\mathbf{D}_{0} | \mathbf{D}_{1})}_{L_{0}} \right]$$
(7)

In this formula, L_T is constant during the training, as it does not depend on any learnable parameters. Both forward and reverse processes are defined by Gaussian distributions, which allows us to compute the terms L_0 and L_{t-1} in closed form:

$$L_{0} = \mathbb{E}_{q} \left[\frac{1}{2\tilde{\delta}^{2}} \| \mathbf{D}_{0} - \mu_{\theta}(\mathbf{p}_{1}, 1) \|^{2} \right] + C_{0}; \ L_{t-1} = \mathbb{E}_{q} \left[\frac{1}{2\tilde{\delta}^{2}} \left\| \frac{1}{\sigma_{t}^{2}} \mathbf{D}_{0} - \mu_{\theta}(\mathbf{p}_{t}, t) \right\|^{2} \right] + C_{t-1},$$
(8)

where C_0 and C_{t-1} are constants that do not depend on parameters θ . This implies that the most direct parameterization of μ_{θ} is a model that predicts \mathbf{D}_0/σ_t^2 , corresponding to the posterior mean of the forward process. However, for practical reasons, we instead parameterize μ_{θ} as g_{θ}/σ_t^2 which ensures that all model outputs are scaled to have the same variance across timesteps.

$$L_{t-1} = \mathbb{E}_q \left[\frac{1}{2\tilde{\delta}^2 \sigma_t^4} \| \mathbf{D}_0 - g_{\theta}(\mathbf{p}_t, t) \|^2 \right] + C_{t-1}, \tag{9}$$

Following [20], we replace L_{t-1} with its simplified version by removing the scaling coefficient $2\tilde{\delta}^2 \sigma_t^4$, resulting in the following loss function:

$$L_{\mathbf{D}}(\theta) = \mathbb{E}_{\mathbf{w}^{y}, t, \mathbf{p}_{t}} \left[\| \mathbf{D}_{0}(\mathbf{E}_{\mathbf{w}^{y}}) - g_{\theta}(\mathbf{p}_{t}, t) \|^{2} \right]$$
(10)

However, this loss function is challenging to optimize due to the high variance and dimensionality of D_0 . To address this issue, we introduce the following theorem:

Theorem 4.1. Let $g^*(\mathbf{p}_t, t)$ be an optimal prediction for Eq. 10. Then $g^*(\mathbf{p}_t, t) = \mathbf{D}_0(f^*(\mathbf{p}_t, t)) + C$, where C is a constant that does not depend on $f^*(\mathbf{p}_t, t)$ and $f^*(\mathbf{p}_t, t)$ is an optimal prediction for Eq. 11.

$$L_{\mathbf{E}}(\theta) = \mathbb{E}_{\mathbf{w}^{y}, t, \mathbf{p}_{t}} \left[\| \mathbf{E}_{\mathbf{w}^{y}} - f_{\theta}(\mathbf{p}_{t}, t) \|^{2} \right]$$
(11)

We train the model f_{θ} by minimizing Eq. 11. During the sampling, we initialize from $\mathbf{D}_{T} \sim \mathcal{N}(0, \tilde{\delta}^{2}I)$ and iteratively update it over 200 steps using the following scheme:

$$\mathbf{D}_{t-1} = \frac{1}{\sigma_{t-1}^2} \mathbf{D}_0(f_{\theta}(\mathbf{p}_t, t)) + \tilde{\delta}\varepsilon, \tag{12}$$

Note that by Th. 4.1, this procedure is equivalent to updating \mathbf{D}_{t-1} as $\mathbf{D}_{t-1} = g_{\theta}(\mathbf{p}_t, t)/\sigma_{t-1}^2 + \tilde{\delta}\varepsilon$, where g_{θ} is optimized with Eq. 10, because models take $\mathbf{p}_t = \operatorname{softmax}(\mathbf{D}_t)$ as input, which is invariant to shifts of \mathbf{D}_t . The proof of Th. 4.1 is provided in Appendix D.

In contrast, related methods such as SSD-LM [16] and TESS [25] employ cross-entropy loss during training. While our method is also compatible with this loss, in our experiments it led to inferior performance and faster overfitting. Therefore, we chose to rely on the MSE objective.

Algorithm 1 Training

```
Input: \mathbf{w}^x, \mathbf{w}^y, \delta, t \sim \mathcal{U}(1, T), \varepsilon \sim \mathcal{N}(0, I)

Compute \mathbf{D}_0 with Eq. 2

Compute \mathbf{D}_t = \mathbf{D}_t/\sigma_t^2 + \delta \varepsilon

Compute \mathbf{p}_t = \operatorname{softmax}(\mathbf{D}_t)

Minimize \|\mathbf{E}_{\mathbf{w}^y} - f_{\theta}(\mathbf{p}_t, t, \mathbf{w}^x)\|^2
```

Algorithm 2 Sampling

```
Input: Source text \mathbf{w}^x, model f_{\theta}, noise std \tilde{\delta} Sample \mathbf{D}_T \sim \mathcal{N}(0, \tilde{\delta}^2 I) for t in \{T, \dots, 1\} do

Compute \mathbf{p}_t = \operatorname{softmax}(\mathbf{D}_t)

Compute \mathbf{D}_{t-1} with Eq. 12 end for

Decode tokens \hat{\mathbf{w}}^y = \operatorname{argmax}(\mathbf{D}_0)
```

4.3 Noise scheduler

The noise scheduler plays a crucial role in the diffusion process by controlling the rate at which the signal decays over time. Following the observation that text diffusion models benefit from adding more noise at the early stages of the forward process [42], we define our noise schedule as follows:

$$\sigma_t = (\sigma_{\text{max}} - \sigma_{\text{min}}) \frac{2}{\pi} \arctan\left(\frac{1}{d}\sqrt{\frac{t}{T - t + \epsilon}}\right) + \sigma_{\text{min}}, \quad \forall t \in [0, T]$$
 (13)

Here, σ_{\min} and σ_{\max} sets the minumum and maximum bandwidth respectively, d controls the rate of noise accumulation, and ϵ is a small constant added to prevent division by zero. We use $\sigma_{\min} = 1.5$, $\sigma_{\max} = 200$ and d = 9 throughout our experiments. Also, we set $\delta = 1$ during the training.

4.4 Sequence length

Because diffusion models operate over fixed-length sequences, we pad all shorter sequences using a special padding token, which the model is trained to predict. To limit computational overhead, we set the maximum sequence length for each dataset to approximately the 99th percentile of training set sequence lengths. The exact values used for each dataset are provided in the Appendix E.

5 Experiments

Implementation details In all experiments, we use a pre-trained embedding matrix \mathbf{E} from the BERT [9] model. We normalize this matrix to have zero mean and unit variance and keep it fixed throughout training. Although the model receives the soft token distribution \mathbf{p}_t as input, it does not operate directly on these distribution. Instead, we compute a weighted average of token embeddings, $\mathbf{p}_t\mathbf{E}$, which yields a lower-dimensional, more tractable representation for the model to process.

Our model architecture is based on the design proposed in [42], consisting of Transformer decoder layers [44] augmented with UNet-style skip connections. Specifically, the output of the first layer is added to the input of the last, the second to the second-last, and so on. The full model has 12 layers and approximately 100M parameters. For conditional generation, we modify the model to accept an input sequence \mathbf{w}^x , which is processed by an additional 6-layer Transformer encoder. The encoder output is integrated into the decoder through cross-attention mechanisms. For timestep conditioning, we adopt the approach from [13], plugging learned timestep embeddings into each Transformer block akin to positional embeddings. The complete set of hyperparameters used for training and evaluation is provided in Appendix E.

Decoder As noted in [42], using a context-aware decoder for embedding-based diffusion improves generation quality. However, in the case of our smoothing diffusion, the decoder type had negligible impact on output quality. Further details can be found in Appendix F. Therefore, in all experiments we decode predicted embeddings by rounding them to closest tokens, which is equivalent to choosing a token with the highest \mathbf{D}_0 value.

5.1 The importance of $\tilde{\delta}$

Before presenting results on seq-to-seq generation tasks, we highlight the importance of the hyperparameter $\tilde{\delta}$, which controls the stochasticity of the denoising process. To illustrate its impact, we evaluate generation quality on an unconditional generation task using different values of $\tilde{\delta}$. Specifically, we use the **ROCStories** dataset and assess performance using three metrics: **perplexity** (to estimate average text quality), **diversity** (to measure lexical variety), and the **MAUVE Score** [36] (to evaluate the overall similarity of generated texts to the reference distribution).

Figure 2 shows the results for a model trained with $\delta=1$. We observe that lower values of $\tilde{\delta}$ lead to better perplexity scores but almost-zero diversity. In other words, reduced stochasticity improves the quality of individual texts but decreases their uniqueness. This trade-off is actually desirable for sequence-to-sequence tasks, where diversity typically arises naturally from the varying input condi-

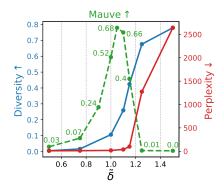


Figure 2: Unconditional generation quality for $\delta = 1$ and varying $\tilde{\delta}$.

tions. Based on this insight, we set $\tilde{\delta} = 0.25$ for all sequence-to-sequence experiments.

In contrast, for unconditional generation, the optimal value of $\tilde{\delta}$ is slightly higher than the one used during training, as indicated by the MAUVE Score. At this point, the generated texts exhibit sufficient diversity while maintaining acceptable perplexity. These findings show that $\tilde{\delta}$ has a strong influence on the generation process and should be tuned carefully depending on the target task.

Datasets We evaluate SMOOTHIE on four datasets of varying difficulty. For Paraphrase Generation, we use the Quora Question Pairs (**QQP**) dataset [6], which contains 147K pairs of semantically equivalent questions. For Question Generation, we adopt the **Quasar-T** dataset [10], processed by [13], resulting in 119K document-question pairs. For Text Simplification, we use the **Newsela-Auto** dataset [24], also sourced from [13], which includes 677K complex sentences from English news articles paired with human-written simplifications. For Summarization, we use the **XSum** dataset [34], comprising 204K BBC articles and their corresponding summaries. More detailed information about each dataset is provided in the Appendix G.

Metrics Following the evaluation protocol from prior work [13, 25], we employ a combination of n-gram-based, diversity and semantic similarity metrics. Specifically, we report **BLEU** [35] and **ROUGE-1/2/L** [28] scores to measure lexical overlap between generated and reference texts, and **BERTScore** (**BS**) [48] to assess semantic similarity. For BERTScore, we use the microsoft/deberta-xlarge-mnli model to ensure consistency with previous studies [13, 47, 25].

To evaluate the diversity of generated texts, we compute n-gram diversity [8], which reports the number of unique unigrams (**Div-1**) and 4-grams (**Div-4**). Additionally, for the text simplification task, we include the **SARI** metric [1], which has been shown to correlate well with human judgment.

Baselines We compare SMOOTHIE against several diffusion-based and autoregressive baselines, all with approximately 100M parameters and trained from scratch on each dataset. The diffusion-based baselines include DiffuSeq [13], SeqDiffuSeq [47], SSD-LM [16], TESS [25], AR-Diffusion [46], and GENIE [29]. For autoregressive baselines, we evaluate BART [26], GPT-2 [38], GPVAE-T5 [11], FLAN-T5 [7], and a standard Transformer model [44]. TESS approach uses pre-trained RoBERTa [30] to initialize their diffusion model. We compare only to the model trained with random initialization for a fair comparison.

Additionally, we conduct a rigorous comparison of our proposed distance-based latent space with two previously explored alternatives: the embedding space [13, 47] and the simplex space [16, 25]. To ensure a fair evaluation, we train all diffusion models under identical conditions, keeping the architecture, training hyperparameters, and decoding strategy fixed. The only variables are the latent space and its associated noise schedule. For embedding-based diffusion, we use the noise scheduler from [42], while for simplex-based diffusion, we adopt the scheduler from [16]. In all three cases, sampling is performed using a procedure defined in the respective latent space, following the formulation in Eq. 12. SMOOTHIE and the embedding-based diffusion model are trained using mean

Table 2: Results on XSum dataset.

	XSum				
Method	BS↑	R-1/2/L ↑			
Transformer [†] FLAN-T5 [†]	— 72.7	30.5/10.4/24.2 34.6/12.9/27.2			
DiffuSeq [♦] AR-Diffusion [♦] GENIE [♦]	46.8 — —	18.9/1.3/13.6 31.7/10.1/24.7 29.3/8.3/21.9			
Embedding* Simplex* SMOOTHIE*	65.6 62.1 68.0	30.8/9.1/23.6 28.7/7.9/22.5 32.9/10.5/25.6			

Table 3: Results on Quasar-T dataset.

	Quasar-T					
Method	BS↑	BLEU ↑	R-L↑	D-1/4		
BART [†]	66.2	17.4	38.8	98.2/61.7		
GPT-2 [†]	60.5	7.4	27.2	96.0/92.2		
GPVAE-T5 [†]	63.1	12.5	33.9	93.8/72.8		
DiffuSeq [♦]	59.4	15.8	_	91.1/—		
SeqDiffuSeq ^{\$}	61.4	17.2	_	92.7/—		
SSD-LM [⋄]	62.8	14.1	38.5	<u>94.5</u> /56.9		
TESS ^{\$\displays (random)}	60.8	<u>19.0</u>	36.1	96.1 /62.4		
Embedding*	62.0	18.0	35.6	92.5/ 64.5		
Simplex*	63.0	19.5	36.9	92.8/63.5		
Smoothie*	62.4	<u>19.0</u>	36.0	92.8/62.5		
$SMOOTHIE^* + SC$	62.4	19.5	36.7	92.0/63.3		

squared error (MSE) loss, while simplex-based diffusion is trained with cross-entropy loss, as it is not suitable for predicting continuous embeddings.

5.2 Results

We now present a numerical comparison of SMOOTHIE with other generative models. In general, SMOOTHIE outperforms each of other text diffusion approaches on the majority of tasks. We take the results of other methods from previous works: for all datasets except XSum, we use results reported in [25]; for XSum, we take DiffuSeq and FLAN-T5 results from [32], and AR-Diffusion, GENIE, and Transformer results from [46]. We re-implement and train the embedding- and simplex-based diffusion baselines within our framework for a fair comparison. For clarity we mark autoregressive methods with †, previous diffusion approaches with \diamond and our implementations with \star .

Summarization Table 2 presents the results for the summarization task on the **XSum** dataset. SMOOTHIE consistently outperforms all diffusion-based baselines, losing only to the autoregressive FLAN-T5 model, which achieves the highest overall performance.

Question generation Table 3 shows the results for the question generation task on the **Quasar-T** dataset. Notably, the simplex-based diffusion combined with our architecture achieves the best overall performance, surpassing even TESS, which uses the same diffusion process. This demonstrates that choosing the right model architecture is no less important than choosing the right diffusion space.

While SMOOTHIE slightly underperforms relative to the simplex-based variant, it still outperforms most other diffusion models and all autoregressive baselines except BART. We also tried to incorporate the self-conditioning technique [5] (+ SC), which has shown significant quality improvements in prior work on text diffusion models [25, 47, 42]. However, we observe only marginal gains. Considering that self-conditioning increases training time by approximately $1.5\times$, we conclude that the trade-off is not justified, and therefore do not include it in our evaluations on other datasets.

Text simplification The results for the text simplification task on the **Newsela-Auto** dataset are shown in Table 4. While we use 200 denoising steps for all other datasets, we increase amount of steps to 500 for **Newsela-Auto**, as this leads to a noticeable improvement in quality (see Sec. 5.3). On this dataset, SMOOTHIE outperforms both the embedding- and simplex-based diffusion models. Interestingly, SeqDiffuSeq and DiffuSeq showed exceptionally well performance, in contrast to their results on the other datasets. Although SMOOTHIE lags behind these two methods, it still significantly outperforms SSD-LM. Among all evaluated models, the autoregressive BART achieves the best overall performance.

Paraphrase generation Table 4 reports the results for paraphrase generation on the **QQP** dataset. On this task, our distance-based latent space clearly outperforms both the embedding- and simplex-based variants. SMOOTHIE achieves results comparable to SSD-LM in terms of BERTScore while significantly outperforming it in BLEU and ROUGE-L. Overall, SMOOTHIE surpasses all autoregressive baselines, falling behind only BART.

Table 4: Results on Newsela-Auto and QQP datasets.

	Newsela-Auto				QQP			
Method	BS↑	BLEU ↑	R-L↑	SARI ↑	BS↑	BLEU ↑	R-L↑	D-1/4
BART [†] GPT-2 [†]	81.7 80.2	41.4 30.8	58.1 54.6	49.9 —	85.7 82.5	30.4 19.8	61.4 52.1	98.8/61.0 98.0/62.5
GPVAE-T5 [†]	81.7	33.9	58.3	_	84.7	24.1	58.9	96.9/61.7
DiffuSeq [♦] SeqDiffuSeq [♦] SSD-LM [♦]	79.1 82.1 69.5	29.9 37.1 12.5	 39.6	_ 36.3	79.5 82.9 83.8	18.5 23.3 22.9	 58.3	97.6/— 98.1/— 98.8 /57.3
Embedding* Simplex* SMOOTHIE*	71.6 69.3 73.7	19.3 16.6 19.9	41.6 39.9 45.2	37.1 35.1 37.6	79.8 80.1 83.1	26.4 25.8 28.3	54.9 53.2 59.3	96.2/ 64.9 96.7/ <u>64.8</u> <u>98.6</u> /59.8

Table 5: Mean rank of methods across Quasar-T, Newsela-Auto and QQP datasets, lower is better.

	DiffuSeq	SeqDiffuSeq	SSD-LM	Embedding	Simplex	SMOOTHIE
Mean Rank ↓	4.67	2.67	4.33	4	3.33	2

Mean rank To assess generalization across tasks, Table 5 reports the mean rank of diffusion models across all datasets except XSum, which is excluded due to limited overlap in evaluated methods. Rankings are computed based on the average of BERTScore and BLEU metrics. SMOOTHIE achieves the best mean rank, while requiring 2 to $10\times$ fewer denoising steps than other methods. Notably, SMOOTHIE outperforms variants that rely on alternative latent space formulations. These results highlight the importance of modeling both the discrete nature of text and the semantic relationships between tokens when designing diffusion processes.

5.3 Amount of denoising steps

Table 6 presents the relationship between the number of denoising steps and the generation quality of SMOOTHIE in terms of BERTScore. We observe that for Quasar-T and QQP, the quality remains largely stable regardless of the number of steps. In contrast, for XSum and Newsela-Auto, performance continues to improve as the number of steps increases. This aligns with the observation made in the TESS paper [25], which suggests that the optimal number of denoising steps correlates with the complexity of the task.

Table 6: BERTScore dependence on the number of denoising steps for different datasets.

Steps XSum		Quasar-T	Newsela-Auto	QQP
50	66.6	62.3	71.9	82.7
100	67.5	62.4	72.8	83.0
200	68.0	62.4	73.1	83.1
500	68.4	62.3	73.7	83.1

6 Conclusion

In this work, we introduce SMOOTHIE, a text diffusion method that constructs its diffusion process with consideration of the discrete nature of text and the semantic relationships between tokens. To capture these properties, each token is mapped to a vector of Euclidean distances between its embedding and the embeddings of all tokens in the vocabulary. Our choice of the Euclidean distance is based on the Euclidean semantic space hypothesis [17], which posits that semantic similarity correlates with Euclidean proximity in embedding space.

Our method also can be applicable to other categorical domains where semantic relationships exist between categories. However, in such cases, a different distance metric more suited to the domain's properties may be required. We leave the exploration of this direction to future work.

Empirical results on four sequence-to-sequence tasks demonstrate that SMOOTHIE outperforms existing text diffusion methods, as well as our diffusion model with alternative diffusion latent spaces that do not rely on additional encoders.

Acknowledgments

We are grateful to Ildus Sadrtdinov for his valuable insights and discussions throughout this project. The paper was supported in part through computational resources of HPC facilities at HSE University.

References

- [1] Optimizing statistical machine translation for text simplification. volume 4, pages 401–415, 2016. URL https://www.aclweb.org/anthology/Q16-1029.
- [2] Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 17981–17993. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/958c530554f78bcd8e97125b70e6973d-Paper.pdf.
- [3] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, Varun Jampani, and Robin Rombach. Stable video diffusion: Scaling latent video diffusion models to large datasets, 2023.
- [4] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models, 2021. URL https://arxiv.org/abs/2012.07805.
- [5] Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. Analog bits: Generating discrete data using diffusion models with self-conditioning, 2023.
- [6] Zihang Chen, Hongbo Zhang, Xiaoji Zhang, and Leqi Zhao. Quora question pairs. 2017. URL https://api.semanticscholar.org/CorpusID:233225749.
- [7] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53, 2024.
- [8] Aditya Deshpande, Jyoti Aneja, Liwei Wang, Alexander G. Schwing, and David Forsyth. Fast, diverse and accurate image captioning guided by part-of-speech. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10687–10696, 2019. doi: 10.1109/CVPR.2019.01095.
- [9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, 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, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology. org/N19-1423.
- [10] Bhuwan Dhingra, Kathryn Mazaitis, and William W Cohen. Quasar: Datasets for question answering by search and reading. arXiv preprint arXiv:1707.03904, 2017.
- [11] Wanyu Du, Jianqiao Zhao, Liwei Wang, and Yangfeng Ji. Diverse text generation via variational encoder-decoder models with gaussian process priors, 2022. URL https://arxiv.org/abs/2204.01227.
- [12] Zach Evans, CJ Carr, Josiah Taylor, Scott H. Hawley, and Jordi Pons. Fast timing-conditioned latent audio diffusion, 2024.
- [13] Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and Lingpeng Kong. Diffuseq: Sequence to sequence text generation with diffusion models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=jQj-_rLVXsj.

- [14] Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and Lingpeng Kong. DiffuSeq-v2: Bridging discrete and continuous text spaces for accelerated Seq2Seq diffusion models. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9868–9875, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.660. URL https://aclanthology.org/2023.findings-emnlp.660/.
- [15] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, Advances in Neural Information Processing Systems, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/file/f033ed80deb0234979a61f95710dbe25-Paper.pdf.
- [16] Xiaochuang Han, Sachin Kumar, and Yulia Tsvetkov. SSD-LM: Semi-autoregressive simplex-based diffusion language model for text generation and modular control. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11575–11596, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023. acl-long.647. URL https://aclanthology.org/2023.acl-long.647.
- [17] Tatsunori B. Hashimoto, David Alvarez-Melis, and Tommi S. Jaakkola. Word embeddings as metric recovery in semantic spaces. *Transactions of the Association for Computational Linguistics*, 4:273–286, 2016. doi: 10.1162/tacl_a_00098. URL https://aclanthology.org/Q16-1020/.
- [18] Zhengfu He, Tianxiang Sun, Qiong Tang, Kuanning Wang, Xuanjing Huang, and Xipeng Qiu. DiffusionBERT: Improving generative masked language models with diffusion models. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4521–4534, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10. 18653/v1/2023.acl-long.248. URL https://aclanthology.org/2023.acl-long.248/.
- [19] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications, 2021. URL https://openreview.net/forum?id=qw8AKxfYbI.
- [20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020.
- [21] Emiel Hoogeboom and Tim Salimans. Blurring diffusion models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=0jDkC57x5sz.
- [22] Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 12454–12465. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/67d96d458abdef21792e6d8e590244e7-Paper.pdf.
- [23] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. ACM Transactions on Information Systems, 43(2):1–55, January 2025. ISSN 1558-2868. doi: 10.1145/3703155. URL http://dx.doi.org/10.1145/3703155.
- [24] Chao Jiang, Mounica Maddela, Wuwei Lan, Yang Zhong, and Wei Xu. Neural crf model for sentence alignment in text simplification. In *Proceedings of the Association for Computational Linguistics (ACL)*, 2020.

- [25] Rabeeh Karimi Mahabadi, Hamish Ivison, Jaesung Tae, James Henderson, Iz Beltagy, Matthew Peters, and Arman Cohan. TESS: Text-to-text self-conditioned simplex diffusion. In Yvette Graham and Matthew Purver, editors, *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2347–2361, St. Julian's, Malta, March 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.eacl-long.144.
- [26] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pretraining for natural language generation, translation, and comprehension. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.703. URL https://aclanthology.org/2020.acl-main.703.
- [27] Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. Diffusion-lm improves controllable text generation. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, Advances in Neural Information Processing Systems, volume 35, pages 4328–4343. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/1be5bc25d50895ee656b8c2d9eb89d6a-Paper-Conference.pdf.
- [28] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.
- [29] Zhenghao Lin, Yeyun Gong, Yelong Shen, Tong Wu, Zhihao Fan, Chen Lin, Nan Duan, and Weizhu Chen. Text generation with diffusion language models: a pre-training approach with continuous paragraph denoise. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.
- [30] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019.
- [31] Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution, 2024. URL https://arxiv.org/abs/2310.16834.
- [32] Justin Lovelace, Varsha Kishore, Chao Wan, Eliot Shekhtman, and Kilian Q Weinberger. Latent diffusion for language generation. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 56998-57025. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/b2a2bd5d5051ff6af52e1ef60aefd255-Paper-Conference.pdf.
- [33] Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. A corpus and cloze evaluation for deeper understanding of commonsense stories. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 839–849, 2016.
- [34] Shashi Narayan, Shay B. Cohen, and Mirella Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1206. URL https://aclanthology.org/D18-1206.
- [35] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In Pierre Isabelle, Eugene Charniak, and Dekang Lin, editors, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://aclanthology.org/P02-1040/.

- [36] Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. Mauve: Measuring the gap between neural text and human text using divergence frontiers. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 4816–4828. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/260c2432a0eecc28ce03c10dadc078a4-Paper.pdf.
- [37] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis, 2023. URL https://arxiv.org/abs/2307.01952.
- [38] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019. URL https://api.semanticscholar.org/CorpusID:160025533.
- [39] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning Volume 37*, ICML'15, page 1530–1538. JMLR.org, 2015.
- [40] Severi Rissanen, Markus Heinonen, and Arno Solin. Generative modelling with inverse heat dissipation. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=4PJUBT9f201.
- [41] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.
- [42] Alexander Shabalin, Viacheslav Meshchaninov, Egor Chimbulatov, Vladislav Lapikov, Roman Kim, Grigory Bartosh, Dmitry Molchanov, Sergey Markov, and Dmitry Vetrov. Tencdm: Understanding the properties of the diffusion model in the space of language model encodings, 2025. URL https://arxiv.org/abs/2402.19097.
- [43] Chengcheng Shao, Giovanni Luca Ciampaglia, Onur Varol, Kai-Cheng Yang, Alessandro Flammini, and Filippo Menczer. The spread of low-credibility content by social bots. *Nature Communications*, 9(1), November 2018. ISSN 2041-1723. doi: 10.1038/s41467-018-06930-7. URL http://dx.doi.org/10.1038/s41467-018-06930-7.
- [44] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- [45] Laura Weidinger, Jonathan Uesato, Maribeth Rauh, Conor Griffin, Po-Sen Huang, John Mellor, Amelia Glaese, Myra Cheng, Borja Balle, Atoosa Kasirzadeh, Courtney Biles, Sasha Brown, Zac Kenton, Will Hawkins, Tom Stepleton, Abeba Birhane, Lisa Anne Hendricks, Laura Rimell, William Isaac, Julia Haas, Sean Legassick, Geoffrey Irving, and Iason Gabriel. Taxonomy of risks posed by language models. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '22, page 214–229, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450393522. doi: 10.1145/3531146.3533088. URL https://doi.org/10.1145/3531146.3533088.
- [46] Tong Wu, Zhihao Fan, Xiao Liu, Hai-Tao Zheng, Yeyun Gong, yelong shen, Jian Jiao, Juntao Li, zhongyu wei, Jian Guo, Nan Duan, and Weizhu Chen. AR-diffusion: Auto-regressive diffusion model for text generation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=0EG6qUQ4xE.
- [47] Hongyi Yuan, Zheng Yuan, Chuanqi Tan, Fei Huang, and Songfang Huang. Seqdiffuseq: Text diffusion with encoder-decoder transformers. *ArXiv*, abs/2212.10325, 2022.
- [48] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert, 2020. URL https://arxiv.org/abs/1904.09675.

A Limitations

Pre-trained Embeddings Our proposed method relies on a pre-trained embedding matrix **E** from the BERT model. While this choice simplifies the training process and improves its stability, it limits the model's scalability and may cap its generation quality, because finetuning embeddings for a specific task should offer better results. An end-to-end training approach, as used in [27, 13, 25], could be applied to our method as well. We leave the exploration of this approach for future work.

Fixed Sequence Length As with most text diffusion models, our method operates with a fixed sequence length. Variable-length outputs are emulated by discarding tokens past the end-of-sequence (EOS) token. This strategy introduces inefficiencies during training and generation, as the model must predict padding tokens regardless of actual sequence length. To the best of our knowledge, dynamically varying sequence lengths during the denoising stage remains an underexplored area. SeqDiffuSeq [47] addresses this by truncating sequences early, based on the observation that the EOS token position often stabilizes early in denoising. However, this is an ad hoc solution, and more advanced approaches need to be developed.

B Societal Impact

Language models have been shown to produce harmful outputs [45], spread disinformation [43], hallucinate [23], and potentially violate user privacy [4]. Although our study focuses on tasks that differ from those typically used in prior harmfulness evaluations, future scaling of our approach could lead to similar negative outcomes. Research on methods for mitigating model harmfulness is actively developing, and we believe that insights from this work may also inform improvements in the reliability and safety of text diffusion models.

C Relationship between distance-based and simplex-based latent spaces

In this section, we demonstrate that our proposed *distance-based latent space* generalizes the *simplex-based latent space*. Specifically, we show that the simplex-based latent space corresponds to a special case of a distance-based latent space when equipped with a trivial metric.

SMOOTHIE maps each token w to a latent vector \mathbf{d}^w , where each component is given by:

$$\mathbf{d}_{(i)}^{w} = -\frac{1}{2} \|\mathbf{E}_{w} - \mathbf{E}_{i}\|^{2}.$$
 (14)

For other categorical domains, the Euclidean distance can be replaced with a more suitable metric $\rho(w,i)$, leading to:

$$\mathbf{d}_{(i)}^{w} = -\rho(w, i). \tag{15}$$

To relate this to simplex-based representations, consider the case where ρ is the *trivial metric*:

$$\rho(w,i) = [w \neq i],\tag{16}$$

i.e., 0 when w = i and 1 otherwise. Under this choice, the latent vector becomes:

$$\mathbf{d}_{(i)}^{w} = \begin{cases} 0, & i = w, \\ -1, & \text{otherwise.} \end{cases}$$
 (17)

In comparison, the simplex-based latent space maps each token w to a vector \mathbf{s}^w in the k-logit simplex:

$$\mathbf{s}_{(i)}^{w} = \begin{cases} +k, & i = w, \\ -k, & \text{otherwise.} \end{cases}$$
 (18)

Both SMOOTHIE and simplex diffusion apply a Gaussian diffusion process to corrupt the latent vector:

$$\mathbf{z}_t = \phi_t \mathbf{z}_0 + \gamma_t \varepsilon, \tag{19}$$

where $\mathbf{z}_0 \in \{\mathbf{d}^w, \mathbf{s}^w\}$ and $\varepsilon \sim \mathcal{N}(0, I)$. To form a model input, the corrupted vector is then transformed into a probability distribution using the softmax function:

$$p_t = \operatorname{softmax}(\mathbf{z}_t). \tag{20}$$

Since the softmax function is invariant to uniform additive shifts, we have:

$$\operatorname{softmax}(\phi_t \mathbf{s}^w + \gamma_t \varepsilon) = \operatorname{softmax}(\phi_t (\mathbf{s}^w - k) + \gamma_t \varepsilon) = \operatorname{softmax}(2k\phi_t \mathbf{d}^w + \gamma_t \varepsilon), \quad (21)$$

where the final equality follows from observing that $s^w - k = 2k\mathbf{d}^w$.

This confirms that the simplex-based latent space is equivalent, up to scaling, to the distance-based latent space under the trivial metric. Hence, the simplex-based representation is a special case within the more general distance-based latent space framework.

D Proof of Theorem 4.1

Proof. We begin by recalling a standard result:

Lemma. The minimum value of the function $\mathbb{E}_{\mathbf{y}} [\|\mathbf{y} - \mathbf{z}\|^2]$ is achieved when $\mathbf{z} = \mathbb{E}[\mathbf{y}]$.

Using this lemma, we obtain:

$$g^*(\mathbf{p}_t, t) = \mathbb{E}_{\mathbf{w}^y}[\mathbf{D}_0(\mathbf{E}_{\mathbf{w}^y})] = \mathbb{E}_{\mathbf{w}^y}\left[-\frac{1}{2}\left\{\|\mathbf{E}_{w_i^y} - \mathbf{E}_j\|^2\right\}_{i,j=1}^{m,V}\right] \quad \text{and} \quad f^*(\mathbf{p}_t, t) = \mathbb{E}_{\mathbf{w}^y}[\mathbf{E}_{\mathbf{w}^y}],$$
(22)

where $\mathbf{w}^y \sim p(\mathbf{w}^y \mid \mathbf{p}_t)$. Since both $g^*(\mathbf{p}_t, t)$ and $f^*(\mathbf{p}_t, t)$ are matrices, without loss of generality we will prove this statement for an arbitrary row i and column j. For brevity, we will define $u = \mathbf{E}_{w_i^y}$ and $v = \mathbf{E}_j$. Then, we need to show that

$$\mathbb{E}_{u}\left[-\frac{1}{2}\|u-v\|^{2}\right] = -\frac{1}{2}\|\mathbb{E}[u]-v\|^{2} + C$$
(23)

Expanding both sides:

$$\mathbb{E}_{u} [||u - v||^{2}] = \mathbb{E}[||u||^{2}] - 2v^{\top} \mathbb{E}[u] + ||v||^{2}$$
$$||\mathbb{E}[u] - v||^{2} = ||\mathbb{E}[u]||^{2} - 2v^{\top} \mathbb{E}[u] + ||v||^{2}$$

Subtracting:

$$\mathbb{E}[\|u\|^2] - \|\mathbb{E}[u]\|^2 = \sum_{k=1}^d \text{Var}(u_k) =: C$$

Thus,

$$\mathbb{E}_{u}\left[-\frac{1}{2}\|u-v\|^{2}\right] = -\frac{1}{2}\|\mathbb{E}[u]-v\|^{2} + \underbrace{-\frac{1}{2}C}_{\text{constant}},$$

where C is a constant independent of $\mathbb{E}[u]$.

Since this holds for all (i, j), the matrix identity holds:

$$g^*(\mathbf{p}_t, t) = \mathbf{D}_0(f^*(\mathbf{p}_t, t)) + \mathbf{C}$$

E Implementation details

The hyperparemeters for training and inference of the models across all datasets are presented in Table 7. We trained our models using two 80 GB NVIDIA A100 GPUs for 15 hours on average. For all the tasks, we save checkpoints every 25,000 steps. We select the best checkpoint by the quality on the development set. During generation we do not apply the clamping trick [27], since it does not improve quality in our experiments. We do not use classifier-free guidance [19] for the same reason.

Table 7: Hyperparemeter values.

Hyperparameter	XSum	Quasar-T	Newsela-Auto	QQP			
Tokenizer	bert-base-cased						
Transformer Layers			12				
Transformer Dim			768				
Self-Attention Heads			12				
Optimizer		Ac	lamW				
Learning Rate		$2 \cdot$	10^{-4}				
β_1, β_2		0.9	9, 0.98				
Warmup steps		5	5000				
LR scheduler		Co	nstant				
Weight decay		(0.01				
Gradient clipping			1				
EMA decay		0.	9999				
Batch size			256				
Training steps	275k	375k	250k	125k			
Max input length	512	100	64	50			
Max target length	64	50	64	50			
Generation steps	200	200	500	200			
$\delta,\widetilde{\delta}$	1, 0.25						
$\sigma_{\min}, \sigma_{\max}, d$		1.5,	, 200, 9				

Table 8: Impact of a complex decoder on generation performance on Quasar-T, Newsela-Auto and QQP datasets.

	Quasar-T		Newsela-Auto		QQP			
Method	BS↑	BLEU ↑	R-L↑	BS↑	BLEU ↑	BS↑	BLEU ↑	R-L↑
Embedding	62.0	18.0	35.6	71.6	19.3	79.8	26.4	54.9
SMOOTHIE	62.4	19.0	36.0	73.7	19.9	83.1	28.3	59.3
Embedding + Dec	62.5	19.1	35.7	71.7	19.4	80.0	26.2	54.7
SMOOTHIE + Dec	62.4	18.9	35.9	73.6	19.8	83.0	28.2	59.2

F Decoder ablation

While our diffusion process enables natural decoding of the generated latents, we also experimented with a more complex transformer-based decoder, following the approach in [42], to evaluate whether it can improve quality by correcting diffusion errors. For the decoder, we employed a 3-layer bidirectional Transformer model trained to decode corrupted embeddings by minimizing the following loss function:

$$-\mathbb{E}_{\mathbf{w}^{y},\varepsilon,\sigma}\log p_{\theta}(\mathbf{w}^{y} \mid \mathbf{E}_{\mathbf{w}^{y}} + \sigma\varepsilon), \tag{24}$$

where $\sigma \sim U[0,0.5]$ controls the level of corruption and $\varepsilon \sim \mathcal{N}(0,I)$ is a random Gaussian noise.

Table 8 compares SMOOTHIE with and without the transformer-based decoder across three datasets: Quasar-T, QQP, and Newsela-Auto. We also include an embedding-based Gaussian diffusion model in the comparison to examine whether the choice of latent space influences the impact of the decoder. The results indicate that the addition of a complex decoder has an overall negligible effect on generation quality. Notably, we observe an improvement for the embedding-based model on Quasar-T in terms of BERTScore, along with minor gains on Quasar-T and Newsela-Auto for other metrics. In contrast, the performance of SMOOTHIE insignificantly declines. We can conclude that complex decoder do not affect the generation quality and should be avoided to prevent model complication.

G Dataset statistics

ROCStories The ROCStories dataset [33] contains 98,161 five-sentence commonsense fictional stories that capture causal and temporal relations between everyday events. It is a widely used small-scale benchmark for unconditional text generation. The dataset is split into 93,161 training instances, 4,000 validation instances, and 1,000 test instances.

XSum The XSum dataset [34] is used for extreme summarization of BBC news articles. Each article covers a diverse range of topics (e.g., sports, politics) and is paired with a single-sentence summary. The dataset is divided into 204,045 training, 11,332 validation, and 11,334 test instances.

Quasar-T Quasar-T [10] is a large-scale dataset for the question generation task. It requires models to comprehend natural language queries and extract answers from a large corpus. The dataset consists of open-domain trivia questions and their corresponding answers, collected from various internet sources. We use the version preprocessed by [13], which includes 116,953 training instances, 2,048 validation instances, and 10,000 test instances.

Newsela-Auto The Newsela-Auto dataset [24] is used for the text simplification task. It contains English news articles rewritten into simpler versions by professional editors. The dataset includes 677,751 training pairs, 2,048 validation pairs, and 5,000 test pairs.

QQP The Quora Question Pairs (QQP) dataset [6] consists of over 400,000 question pairs from the Quora platform, each annotated with a binary label indicating whether the two questions are paraphrases. For the paraphrase generation task, we use the subset containing 149,263 positively labeled pairs, split into 119,410 training instances, 14,926 validation instances, and 14,927 test instances.