Diffusion-LM Improves Controllable Text Generation

Xiang Lisa Li
Stanford University
xlisali@stanford.edu

John Thickstun
Stanford University
jthickst@stanford.edu

Ishaan Gulrajani
Stanford Univeristy
igul@stanford.edu

Percy Liang
Stanford Univeristy
pliang@cs.stanford.edu

Tatsunori B. Hashimoto
Stanford Univeristy
thashim@stanford.edu

Abstract

Controlling the behavior of language models (LMs) without re-training is a major open problem in natural language generation. While recent works have demonstrated successes on controlling simple sentence attributes (e.g., sentiment), there has been little progress on complex, fine-grained controls (e.g., syntactic structure). To address this challenge, we develop a new *non-autoregressive* language model based on *continuous* diffusions that we call Diffusion-LM. Building upon the recent successes of diffusion models in continuous domains, Diffusion-LM iteratively denoises a sequence of Gaussian vectors into word vectors, yielding a sequence of intermediate latent variables. The continuous, hierarchical nature of these intermediate variables enables a simple gradient-based algorithm to perform complex, controllable generation tasks. We demonstrate successful control of Diffusion-LM for six challenging fine-grained control tasks, significantly outperforming prior work.[1]

1 Introduction

Large autoregressive language models (LMs) are capable of generating high quality text [33, 3, 5, 45], but in order to reliably deploy these LMs in real world applications, the text generation process needs to be *controllable*: we need to generate text that satisfies desired requirements (e.g. topic, syntactic structure). A natural approach for controlling a LM would be to fine-tune the LM using supervised data of the form (control, text) [16]. However, updating the LM parameters for each control task can be expensive and does not allow for compositions of multiple controls (e.g. generate text that is both positive sentiment *and* non-toxic). This motivates light-weight and modular plug-and-play approaches [6] that keep the LM frozen and steer the generation process using an external classifier that measures how well the generated text satisfies the control. But steering a frozen autoregressive LM has been shown to be difficult, and existing successes have been limited to simple, attribute-level controls (e.g., sentiment or topic) [6, 22, 44].

In order to tackle more complex controls, we propose *Diffusion-LM*, a new language model based on *continuous* diffusions. Diffusion-LM starts with a sequence of Gaussian noise vectors and incrementally denoises them into vectors corresponding to words, as shown in Figure 1. These gradual denoising steps produce a hierarchy of continuous latent representations. We find that this hierarchical and continuous latent variable enables simple, gradient-based methods to perform complex control tasks such as constraining the parse tree of a generated sequence.

Continuous diffusion models have been extremely successful in vision and audio domains [12, 21, 34, 8, 4], but they have not been applied to text because of the inherently discrete nature of text

[1] Code is available at `https://github.com/XiangLi1999/Diffusion-LM.git`

Preprint. Under review.

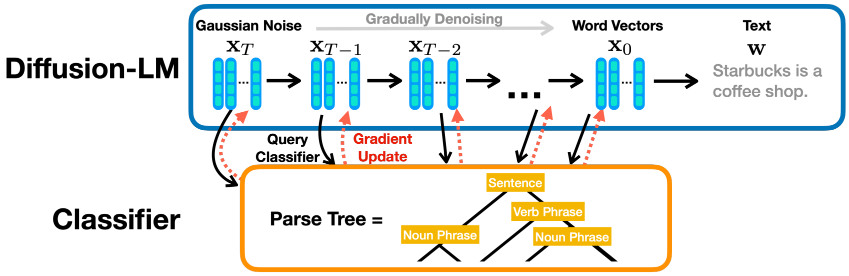

Figure 1: Diffusion-LM iteratively denoises a sequence of Gaussian vectors into word vectors, yielding a intermediate latent variables of decreasing noise level $\mathbf{x}_T \cdots \mathbf{x}_0$. For controllable generation, we iteratively perform gradient updates on these continuous latents to optimize for fluency (parametrized by Diffusion-LM) and satisfy control requirements (parametrized by a classifier).

(§3). Adapting this class of models to text requires several modifications to standard diffusions: we add an embedding step and a rounding step to the standard diffusion process, design a training objective to learn the embedding, and propose techniques to improve rounding (§4). We control Diffusion-LM using a gradient-based method, as shown in Figure 1. This method enables us to steer the text generation process towards outputs that satisfy target structural and semantic controls. It iteratively performs gradient updates on the continuous latent variables of Diffusion-LM to balance fluency and control satisfaction (§5.1).

To demonstrate control of Diffusion-LM, we consider six control targets ranging from fine-grained attributes (e.g., semantic content) to complex structures (e.g., parse trees). Our method almost doubles the success rate of previous plug-and-play methods and matches or outperforms the fine-tuning oracle on all these classifier-guided control tasks (§7.1). In addition to these individual control tasks, we show that we can successfully compose multiple classifier-guided controls to generate sentences with both desired semantic content and syntactic structure (§7.2). Finally, we consider span-anchored controls, such as length control and infilling. Diffusion-LM allows us to perform these control tasks *without* a classifier, and our Diffusion-LM significantly outperforms prior plug-and-play methods and is on-par with an autoregressive LM trained from scratch for the infilling task (§7.3).

2 Related Work

Diffusion Models for Text. Diffusion models [39] have demonstrated great success in continuous data domains [12, 27, 21, 25], producing images and audio that have state-of-the-art sample quality. To handle discrete data, past works have studied text diffusion models on *discrete* state spaces, which defines a corruption process on discrete data (e.g., each token has some probability to be corrupted to an absorbing or random token) [1, 14, 15]. In this paper, we focus on *continuous* diffusion models for text and to the best of our knowledge, our work is the first to explore this setting. In contrast to discrete diffusion LMs, our continuous diffusion LMs induce continuous latent representations, which enables efficient gradient-based methods for controllable generation.

Autoregressive and Non-autoregressive LMs. Most large pre-trained LMs are left-to-right autoregressive (e.g., GPT-3 [3], PaLM [5]). The fixed generation order limits the models' flexibility in many controllable generation settings, especially those that impose controls globally on both left and right contexts. One example is infilling, which imposes lexical control on the right contexts; another example is syntactic structure control, which controls global properties involving both left and right contexts. Since autoregressive LMs cannot directly condition on right contexts, prior works have developed specialized training and decoding techniques for these tasks [38, 9, 30]. For example, Qin et al. [31] proposed a decoding method that relaxes the discrete LM outputs to continuous variables and backpropagates gradient information from the right context. Diffusion-LM can condition on arbitrary classifiers that look at complex, global properties of the sentence. There are other non-autoregressive LMs that have been developed for machine translation and speech-to-text tasks [11, 37]. However these methods are specialized for speech and translation settings, where the entropy over valid outputs is low, and it has been shown that these approaches fail for language modeling [35].

Plug-and-Play Controllable Generation. Plug-and-play controllable generation aims to keep the LM frozen and steer its output using potential functions (e.g., classifiers). Given a probabilistic potential function that measures how well the generated text satisfies the desired control, the generated text should be optimized for both control satisfaction (measured by the potential function) and fluency (measured by LM probabilities) . There are several plug-and-play approaches based on autoregressive LMs: FUDGE [44] reweights the LM prediction at each token with an estimate of control satisfaction for the partial sequence; GeDi [22] and DExperts [24] reweight the LM prediction at each token with a smaller LM finetuned/trained for the control task.

The closest work to ours is PPLM [6], which runs gradient ascent on an autoregressive LM's hidden activations to steer the next token to satisfy the control and maintain fluency. Because PPLM is based on autoregressive LMs, it can only generate left-to-right. This prevents PPLM from repairing and recovering errors made in previous generation steps. Despite their success on attribute (e.g., topic) controls, we will show these plug-and-play methods for autoregressive LMs fail on more complex control tasks such as controlling syntactic structure and semantic content in §7.1. We demonstrate that Diffusion-LM is capable of plug-and-play controllable generation by applying classifier-guided gradient updates to the continuous sequence of latent variables induced by the Diffusion-LM.

3 Problem Statement and Background

We first define controllable generation (§3.1) and then review continuous diffusion models (§3.3).

3.1 Generative Models and Controllable Generation for Text

Text generation is the task of sampling $\mathbf{w}$ from a trained language model $p_{\text{lm}}(\mathbf{w})$, where $\mathbf{w} = [w_1 \cdots w_n]$ is a sequence of discrete words and $p_{\text{lm}}(\mathbf{w})$ is a probability distribution over sequences of words. Controllable text generation is the task of sampling $\mathbf{w}$ from a conditional distribution $p(\mathbf{w} \mid \mathbf{c})$, where $\mathbf{c}$ denotes a *control* variable. For syntactic control, $\mathbf{c}$ can be a target syntax tree (Figure 1), while for sentiment control, $\mathbf{c}$ could be a desired sentiment label. The goal of controllable generation is to generate $\mathbf{w}$ that satisfies the control target $\mathbf{c}$.

Consider the plug-and-play controllable generation setting: we are given a language model $p_{\text{lm}}(\mathbf{w})$ trained from a large amount of unlabeled text data, and for each control task, we are given a classifier $p(\mathbf{c} \mid \mathbf{w})$ trained from smaller amount of labeled text data (e.g., for syntactic control, the classifier is a probabilistic parser). The goal is to utilize these two models to approximately sample from the posterior $p(\mathbf{w} \mid \mathbf{c})$ via Bayes rule $p(\mathbf{w} \mid \mathbf{c}) \propto p_{\text{lm}}(\mathbf{w}) \cdot p(\mathbf{c} \mid \mathbf{w})$. Here, $p_{\text{lm}}(\mathbf{w})$ encourages $\mathbf{w}$ to be fluent, and the $p(\mathbf{c} \mid \mathbf{w})$ encourages $\mathbf{w}$ to fulfill the control.

3.2 Autoregressive Language Models

The canonical approach to language modeling factors p_{lm} in an autoregressive left-to-right mannar, $p_{\text{lm}}(\mathbf{w}) = p_{\text{lm}}(w_1) \prod_{i=2}^{n} p_{\text{lm}}(x_i \mid x_{<i})$. In this case, text generation is reduced to the task of repeatedly predicting the next token conditioned on the partial sequence generated so far. The next token prediction $p_{\text{lm}}(x_i \mid x_{<i})$ is often parametrized by Transformer architecture [42].

3.3 Diffusion Models for Continuous Domains

A diffusion model [12, 27] is a latent variable model that models the data $\mathbf{x}_0 \in \mathbb{R}^d$ as a Markov chain $\mathbf{x}_T \ldots \mathbf{x}_0$ with each variable in $\mathbb{R}^d$, and $\mathbf{x}_T$ is a Gaussian. The diffusion model incrementally denoises the sequence of latent variables $\mathbf{x}_{T:1}$ to approximate samples from the target data distribution (Figure 2). The initial state $p_\theta(\mathbf{x}_T) \approx \mathcal{N}(0, \mathbf{I})$, and each denoising transition $\mathbf{x}_t \rightarrow \mathbf{x}_{t-1}$ is parametrized by the model $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t))$. For example, μ_θ and Σ_θ may be computed by a U-Net or a Tranformer.

To train the diffusion model, we define a forward process that constructs the intermediate latent variables $\mathbf{x}_{1:T}$. The forward process incrementally adds Gaussian noise to data $\mathbf{x}_0$ until, at diffusion step T, samples $\mathbf{x}_T$ are approximately Gaussian. Each transition $\mathbf{x}_{t-1} \rightarrow \mathbf{x}_t$ is parametrized by $q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1 - \beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I})$, where the hyperparameter β_t is the amount of noise added at diffusion step t. This parametrization of the forward process q contains no trainable parameters and

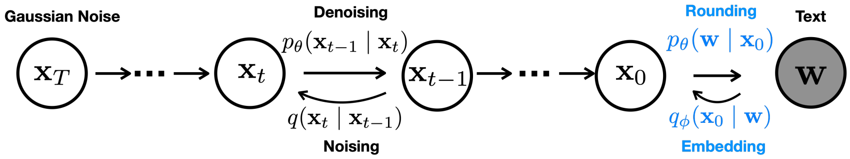
Gaussian Noise
$\mathbf{x}_T$
$\mathbf{x}_t$
Denoising
$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$
$\mathbf{x}_{t-1}$
$q(\mathbf{x}_t \mid \mathbf{x}_{t-1})$
Noising
$\mathbf{x}_0$
Rounding
$p_\theta(\mathbf{w} \mid \mathbf{x}_0)$
$q_\phi(\mathbf{x}_0 \mid \mathbf{w})$
Embedding
Text
$\mathbf{w}$

Figure 2: A graphical model representing the forward and reverse diffusion processes. In addition to the original diffusion models [12], we add a Markov transition between $\mathbf{x}_0$ and $\mathbf{w}$, and propose the embedding §4.1 and rounding §4.2 techniques.

allows us to define a training objective that involves generating noisy data according to a pre-defined forward process q and training a model to reverse the process and reconstruct the data.

The diffusion model is trained to maximize the marginal likelihood of the data $\mathbb{E}_{\mathbf{x}_0 \sim p_{\text{data}}}[\log p_\theta(\mathbf{x}_0)]$, and the canonical objective is the variational lower bound of $\log p_\theta(\mathbf{x}_0)$ [39],

$$\mathcal{L}_{\text{vlb}}(\mathbf{x}_0) = \underset{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{\mathbb{E}} \left[\log \frac{q(\mathbf{x}_T|\mathbf{x}_0)}{p_\theta(\mathbf{x}_T)} + \sum_{t=2}^{T} \log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_0, \mathbf{x}_t)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} - \log p_\theta(\mathbf{x}_0|\mathbf{x}_1) \right]. \tag{1}$$

However, this objective can be unstable and require many optimization tricks to stabilize [27]. To circumvent this issue, Ho et al. [12] devised a simple surrogate objective that expands and reweights each KL-divergence term in $\mathcal{L}_{\text{vlb}}$ to obtain a mean-squared error loss (derivation in Appendix E) which we will refer to as

$$\mathcal{L}_{\text{simple}}(\mathbf{x}_0) = \sum_{t=1}^{T} \underset{q(\mathbf{x}_t|\mathbf{x}_0)}{\mathbb{E}} ||\mu_\theta(\mathbf{x}_t, t) - \hat{\mu}(\mathbf{x}_t, \mathbf{x}_0)||^2,$$

where $\hat{\mu}(\mathbf{x}_t, \mathbf{x}_0)$ is the mean of the posterior $q(\mathbf{x}_{t-1}|\mathbf{x}_0, \mathbf{x}_t)$ which is a closed from Gaussian, and $\mu_\theta(\mathbf{x}_t, t)$ is the predicted mean of $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$ computed by a neural network. While $\mathcal{L}_{\text{simple}}$ is no longer a valid lower bound, prior work has found that it empirically made training more stable and improved sample quality[2]. We will make use of similar simplifications in Diffusion-LM to stabilize training and improve sample quality (§4.1).

4 Diffusion-LM: Continuous Diffusion Language Modeling

Constructing Diffusion-LM requires several modifications to the standard diffusion model. First, we must define an embedding function that maps discrete text into a continuous space. To address this, we propose an end-to-end training objective for learning embeddings (§4.1). Second, we require a rounding method to map vectors in embedding space back to words. To address this, we propose training and decoding time methods to facilitate rounding (§4.2).

4.1 End-to-end Training

To apply a continuous diffusion model to discrete text, we define an embedding function $\textsc{Emb}(w_i)$ that maps each word to a vector in $\mathbb{R}^d$. We define the embedding of a sequence $\mathbf{w}$ of length n to be: $\textsc{Emb}(\mathbf{w}) = [\textsc{Emb}(w_1), \ldots, \textsc{Emb}(w_n)] \in \mathbb{R}^{nd}$.

We propose a modification of the diffusion model training objective (Equation 1) that jointly learns the diffusion model's parameters and word embeddings. In preliminary experiments, we explored random Gaussian embeddings, as well as pre-trained word embeddings [29, 33]. We found that these fixed embeddings are suboptimal for Diffusion-LM compared to end-to-end training[3].

As shown in Figure 2, our approach adds a Markov transition from discrete words $\mathbf{w}$ to $\mathbf{x}_0$ in the forward process, parametrized by $q_\phi(\mathbf{x}_0|\mathbf{w}) = \mathcal{N}(\textsc{Emb}(\mathbf{w}), \sigma_0 I)$. In the reverse process, we add a trainable rounding step, parametrized by $p_\theta(\mathbf{w} \mid \mathbf{x}_0) = \prod_{i=1}^{n} p_\theta(w_i \mid x_i)$, where $p_\theta(w_i \mid x_i)$ is a

[2] Our definition of $\mathcal{L}_{\text{simple}}$ here uses a different parametrization from Ho et al. [12]. We define our squared loss in terms of $\mu_\theta(\mathbf{x}_t, t)$ while they express it in terms of $\epsilon_\theta(\mathbf{x}_t, t)$.

[3] While trainable embeddings perform best on control and generation tasks, we found that fixed embeddings onto the vocabulary simplex were helpful when optimizing for held-out perplexity. We leave discussion of this approach and perplexity results to Appendix F as the focus of this work is generation quality and not perplexity.

softmax distribution. The training objectives introduced in §3 now becomes

$$\begin{aligned}\mathcal{L}^{\text{e2e}}_{\text{vlb}}(\mathbf{w}) &= \underset{q_\phi(\mathbf{x}_0|\mathbf{w})}{\mathbb{E}} \left[\mathcal{L}_{\text{vlb}}(\mathbf{x}_0) + \log q_\phi(\mathbf{x}_0|\mathbf{w}) - \log p_\theta(\mathbf{w}|\mathbf{x}_0)\right], \\ \mathcal{L}^{\text{e2e}}_{\text{simple}}(\mathbf{w}) &= \underset{q_\phi(\mathbf{x}_{0:T}|\mathbf{w})}{\mathbb{E}} \left[\mathcal{L}_{\text{simple}}(\mathbf{x}_0) + ||\text{EMB}(\mathbf{w}) - \mu_\theta(\mathbf{x}_1, 1)||^2 - \log p_\theta(\mathbf{w}|\mathbf{x}_0)\right].\end{aligned} \tag{2}$$

We derive $\mathcal{L}^{\text{e2e}}_{\text{simple}}(\mathbf{w})$ from $\mathcal{L}^{\text{e2e}}_{\text{vlb}}(\mathbf{w})$ following the simplification in §3.3 and our derivation details are in Appendix E. Since we are training the embedding function, q_ϕ now contains trainable parameters and we use the reparametrization trick [36, 18] to backpropagate through this sampling step. Empirically, we find the learned embeddings cluster meaningfully: words with the same part-of-speech tags (syntactic role) tend to be clustered, as shown in Figure 3.

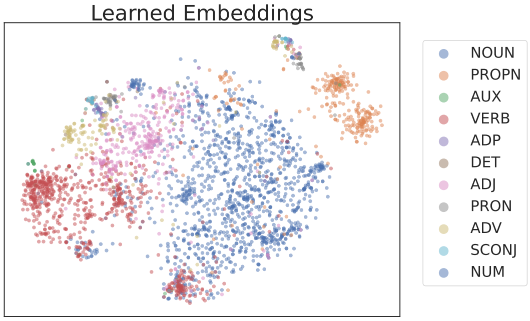

Figure 3: A t-SNE [41] plot of the learned word embeddings. Each word is colored by its POS.

4.2 Reducing Rounding Errors

The learned embeddings define a mapping from discrete text to the continuous $\mathbf{x}_0$. We now describe the inverse process of rounding a predicted $\mathbf{x}_0$ back to discrete text. Rounding is achieved by choosing the most probable word for each position, according to argmax $p_\theta(\mathbf{w} \mid \mathbf{x}_0) = \prod_{i=1}^n p_\theta(w_i \mid x_i)$. Ideally, this argmax-rounding would be sufficient to map back to discrete text, as the denoising steps should ensure that $\mathbf{x}_0$ lies exactly on the embedding of some word. However, empirically, the model fails to generate $\mathbf{x}_0$ that commits to a single word.

One explanation for this phenomenon is that the $\mathcal{L}_{\text{simple}}(\mathbf{x}_0)$ term in our objective 2 puts insufficient emphasis on modeling the structure of $\mathbf{x}_0$. Recall that we defined $\mathcal{L}_{\text{simple}}(\mathbf{x}_0) = \sum_{t=1}^T \mathbb{E}_{\mathbf{x}_t} ||\mu_\theta(\mathbf{x}_t, t) - \hat{\mu}(\mathbf{x}_t, \mathbf{x}_0)||^2$, where our model $\mu_\theta(\mathbf{x}_t, t)$ directly predicts the mean of $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$ for each denoising step t. In this objective, the constraint that $\mathbf{x}_0$ has to commit to a single word embedding will only appear in the terms with t near 0, and we found that this parametrization required careful tuning to force the objective to emphasize those terms (see Appendix H).

Our approach re-parametrizes $\mathcal{L}_{\text{simple}}$ to force Diffusion-LM to explicitly model $\mathbf{x}_0$ in *every* term of the objective. Specifically, we derive an analogue to $\mathcal{L}_{\text{simple}}$ which is parametrized via $\mathbf{x}_0$, $\mathcal{L}^{\text{e2e}}_{\mathbf{x}_0\text{-simple}}(\mathbf{x}_0) = \sum_{t=1}^T \mathbb{E}_{\mathbf{x}_t} ||f_\theta(\mathbf{x}_t, t) - \mathbf{x}_0||^2$, where our model $f_\theta(\mathbf{x}_t, t)$ predicts $\mathbf{x}_0$ directly [4]. This forces the neural network to predict $\mathbf{x}_0$ in every term and we found that models trained with this objective quickly learn that $\mathbf{x}_0$ should precisely centered at a word embedding.

We described how re-parametrization can be helpful for model training, but we also found that the same intuition could be used at decoding time in a technique that we call the *clamping* trick. In the standard generation approach for a $\mathbf{x}_0$-parametrized model, the model denoises $\mathbf{x}_t$ to $\mathbf{x}_{t-1}$ by first computing an estimate of $\mathbf{x}_0$ via $f_\theta(\mathbf{x}_t, t)$ and then sampling $\mathbf{x}_{t-1}$ conditioned on this estimate: $\mathbf{x}_{t-1} = \sqrt{\bar{\alpha}} f_\theta(\mathbf{x}_t, t) + \sqrt{1 - \bar{\alpha}}\epsilon$, where $\bar{\alpha}_t = \prod_{s=0}^t (1 - \beta_s)$ and $\epsilon \sim \mathcal{N}(0, I)$ [5]. In the clamping trick, the model additionally maps the predicted vector $f_\theta(\mathbf{x}_t, t)$ to its nearest word embedding sequence. Now, the sampling step becomes $\mathbf{x}_{t-1} = \sqrt{\bar{\alpha}} \cdot \text{Clamp}(f_\theta(\mathbf{x}_t, t)) + \sqrt{1 - \bar{\alpha}}\epsilon$. The clamping trick forces the predicted vector to commit to a word for intermediate diffusion steps, making the vector predictions more precise and reducing rounding errors.[6]

[4] Predicting $\mathbf{x}_0$ and $\mathbf{x}_{t-1}$ is equivalent up to scaling constants as the distribution of $\mathbf{x}_{t-1}$ can be obtained in closed form via the forward process $\mathbf{x}_{t-1} = \sqrt{\bar{\alpha}}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}}\epsilon$, see Appendix E for further details.

[5] This follows from the marginal distribution $q(\mathbf{x}_t \mid \mathbf{x}_0)$, which is a closed form Gaussian since all the Markov transitions are Gaussian.

[6] Intuitively, applying the clamping trick to early diffusion steps with t near T may be sub-optimal, because the model hasn't figured out what words to commit to. Empirically, applying clamping trick for all diffusion steps doesn't hurt the performance much. But to follow this intuition, one could also set the starting step of the clamping trick as a hyperparameter.

5 Decoding and Controllable Generation with Diffusion-LM

Having described the Diffusion-LM, we now consider the problem of controllable text generation (§5.1) and decoding (§5.2).

5.1 Controllable Text Generation

We now describe a procedure that enables plug-and-play control on Diffusion-LM. Our approach to control is inspired by the Bayesian formulation in §3.1, but instead of performing control directly on the discrete text, we perform control on the sequence of continuous latent variables $\mathbf{x}_{0:T}$ defined by Diffusion-LM, and apply the rounding step to convert these latents into text.

Controlling $\mathbf{x}_{0:T}$ is equivalent to decoding from the posterior $p(\mathbf{x}_{0:T}|\mathbf{c}) = \prod_{t=1}^{T} p(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{c})$, and we decompose this joint inference problem to a sequence of control problems at each diffusion step: $p(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{c}) \propto p(\mathbf{x}_{t-1} \mid \mathbf{x}_t) \cdot p(\mathbf{c} \mid \mathbf{x}_{t-1}, \mathbf{x}_t)$. We further simplify $p(\mathbf{c} \mid \mathbf{x}_{t-1}, \mathbf{x}_t) = p(\mathbf{c} \mid \mathbf{x}_{t-1})$ via conditional independence assumptions from prior work on controlling diffusions [40]. Consequently, for the t-th step, we run gradient update on $\mathbf{x}_{t-1}$:

$$\nabla_{\mathbf{x}_{t-1}} \log p(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{c}) = \nabla_{\mathbf{x}_{t-1}} \log p(\mathbf{x}_{t-1} \mid \mathbf{x}_t) + \nabla_{\mathbf{x}_{t-1}} \log p(\mathbf{c} \mid \mathbf{x}_{t-1}),$$

where both $\log p(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$ and $\log p(\mathbf{c} \mid \mathbf{x}_{t-1})$ are differentiable: the first term is parametrized by Diffusion-LM, and the second term is parametrized by a neural network classifier.

Similar to work in the image setting [8, 40], we train the classifier on the diffusion latent variables and run gradient updates on the latent space $\mathbf{x}_{t-1}$ to steer it towards fulfilling the control. These image diffusion works take one gradient step towards $\nabla_{\mathbf{x}_{t-1}} \log p(\mathbf{c} \mid \mathbf{x}_{t-1})$ per diffusion steps. To improve performance on text and speed up decoding, we introduce two key modifications: fluency regularization and multiple gradient steps.

To generate fluent text, we run gradient updates on a control objective with *fluency regularization*: $\lambda \log p(\mathbf{x}_{t-1} \mid \mathbf{x}_t) + \log p(\mathbf{c} \mid \mathbf{x}_{t-1})$, where λ is a hyperparameter that trades off fluency (the first term) and control (the second term). While existing controllable generation methods for diffusions do not include the $\lambda \log p(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$ term in the objective, we found this term to be instrumental for generating fluent text. The resulting controllable generation process can be viewed as a stochastic decoding method that balances maximizing and sampling $p(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{c})$, much like popular text generation techniques such as nucleus sampling [13] or sampling with low temperature. In order to improve the control quality, we take multiple gradient steps for each diffusion step: we run 3 steps of the Adagrad [7] [10] update for each diffusion steps. To mitigate for the increased computation cost, we downsample the diffusion steps from 2000 to 200, which speeds up our controllable generation algorithm without hurting sample quality much.

5.2 Minimum Bayes Risk Decoding

Many conditional text generation tasks require a *single* high-quality output sequence, such as machine translation or sentence infilling. In these settings, we apply Minimum Bayes Risk (MBR) decoding [23] to aggregate a set of samples $\mathcal{S}$ drawn from the Diffusion-LM , and select the sample that achieves the minimum expected risk under a loss function $\mathcal{L}$ (e.g., negative BLEU score): $\hat{\mathbf{w}} = \operatorname{argmin}_{\mathbf{w} \in S} \sum_{\mathbf{w}' \in S} \frac{1}{|S|} \mathcal{L}(\mathbf{w}, \mathbf{w}')$. We found that MBR decoding often returned high quality outputs, since a low quality sample would be dissimilar from the remaining samples and penalized by the loss function.

6 Experimental Setup

With the above improvements on training (§4) and decoding (§5), we train Diffusion-LM for two language modeling tasks. We then apply the controllable generation method to 5 classifier-guided control tasks, and apply MBR decoding to a classifier-free control task (i.e. infilling).

[7] We tried ablations that replaced Adagrad with SGD, but we found Adagrad to be substantially less sensitive to hyperparameter tuning.

input (Semantic Content) output text	food : Japanese Browns Cambridge is good for Japanese food and also children friendly near The Sorrento .
input (Parts-of-speech) output text	PROPN AUX DET ADJ NOUN NOUN VERB ADP DET NOUN ADP DET NOUN PUNCT Zizzi is a local coffee shop located on the outskirts of the city .
input (Syntax Tree) output text	(TOP (S (NP (*) (*) (*)) (VP (*) (NP (NP (*) (*)))))) The Twenty Two has great food
input (Syntax Spans) output text	(7, 10, VP) Wildwood pub serves multicultural dishes and is ranked 3 stars
input (Length) output text	14 Browns Cambridge offers Japanese food located near The Sorrento in the city centre .
input (left context) input (right context) output text	My dog loved tennis balls. My dog had stolen every one and put it under there. One day, I found all of my lost tennis balls underneath the bed.

Table 1: Example input control and output text for each control tasks.

6.1 Datasets and Hyperparameters

We train Diffusion-LM on two datasets: E2E [28] and ROCStories [26]. The E2E dataset consists of 50K restaurant reviews labeled by 8 fields including food type, price, and customer rating. The ROCStories dataset consists of 98K five-sentence stories, capturing a rich set of causal and temporal commonsense relations between daily events. This dataset is more challenging to model than E2E, because the stories contain a larger vocabulary of 11K words and more diverse semantic content.

Our Diffusion-LM is based on Transformer [42] architecture with 80M parameters, with a sequence length $n = 64$, diffusion steps $T = 2000$ and a square-root noise schedule (see Appendix A for details). We treat the embedding dimension as a hyperparameter, setting $d = 16$ for E2E and $d = 128$ for ROCStories. See Appendix B for hyperparameter details. At decoding time, we downsample to 200 diffusion steps for E2E and maintain 2000 steps for ROCStories. Decoding Diffusion-LM for 200 steps is still 7x slower than decoding autoregressive LMs. For controllable generation, our method based on Diffusion-LM is 1.5x slower than FUDGE but 60x faster than PPLM.

6.2 Control tasks

We consider 6 control tasks shown in Table 1: the first 4 tasks rely on a classifier, and the last 2 tasks are classifier free[8]. For each control task (e.g. semantic content), we sample 200 control targets $\mathbf{c}$ (e.g., rating=5 star) from the validation splits, and we generate 50 samples for each control target. To evaluate the fluency of the generated text, we follow the prior works [44, 6] and feed the generated text to a teacher LM (i.e., a carefully fine-tuned GPT-2 model) and report the perplexity of generated text under the teacher LM. We call this metric lm-score (denoted as lm): a lower lm-score indicates better sample quality. [9] We define success metrics for each control task as follows:

Semantic Content. Given a field (e.g., rating) and value (e.g., 5 star), generate a sentence that covers field=value, and report the success rate by exact match of 'value'.

Parts-of-speech. Given a sequence of parts-of-speech (POS) tags (e.g., *Pronoun Verb Determiner Noun*), generate a sequence of words of the same length whose POS tags (under an oracle POS tagger) match the target (e.g., *I ate an apple*). We quantify success via word-level exact match.

Syntax Tree. Given a target syntactic parse tree (see Figure 1), generate text whose syntactic parse matches the given parse. To evaluate the success, we parse the generated text by an off-the-shelf parser [20], and report F1 scores.

Syntax Spans. Given a target (span, syntactic category) pair, generate text whose parse tree over span $[i, j]$ matches the target syntactic category (e.g. prepositional phrase).We quantify success via the fraction of spans that match exactly.

[8] Length is classifier-free for our Diffusion-LM based methods, but other methods still require a classifier.

[9] Prior works [44, 6] use GPT [32] as the teacher LM whereas we use a fine-tuned GPT-2 model because our base autoregressive LM and Diffusion-LM both generate UNK tokens, which does not exist in pretrained vocabularies of GPT.

Length. Given a target length $10, \ldots, 40$, generate a sequence with a length within ± 2 of the target. In the case of Diffusion-LM, we treat this as a classifier-free control task.

Infilling. Given a left context (O_1) and a right context (O_2) from the aNLG dataset [2], and the goal is to generate a sentence that logically connects O_1 and O_2. For evaluation, we report both automatic and human evaluation from the Genie leaderboard [17].

6.3 Classifier-Guided Control Baselines

For the first 5 control tasks, we compare our method with PPLM, FUDGE, and a fine-tuning oracle. Both PPLM and FUDGE are plug-and-play controllable generation approaches based on an autoregressive LM, which we train from scratch using the GPT-2 small architecture [33].

PPLM[6]. This method runs gradient ascent on the LM activations to increase the classifier probabilities and language model probabilities, and has been successful on simple attribute control. We apply PPLM to control semantic content, but not the remaining 4 tasks which require positional information, as PPLM's classifier lacks positional information.

FUDGE[44]. For each control task, FUDGE requires a future discriminator that takes in a prefix sequence and predicts whether the complete sequence would satisfy the constraint. At decoding time, FUDGE reweights the LM prediction by the discriminator scores.

FT. For each control task, we fine-tune GPT-2 on (control, text) pair, yielding an *oracle* conditional language model that's not plug-and-play. We report both the sampling (with temperature 1.0) and beam search (with beam size 4) outputs of the fine-tuned models, denoted as FT-sample and FT-search.

6.4 Infilling Baselines

We compare to 3 specialized baseline methods developed in past work for the infilling task.

DELOREAN [30]. This method continuously relaxes the output space of a left-to-right autoregressive LM, and iteratively performs gradient updates on the continuous space to enforce fluent connection to the right contexts. This yields a continuous vector which is rounded back to text.

COLD[31]. COLD specifies an energy-based model that includes fluency (from left-to-right and right-to-left LM) and coherence constraints (from lexical overlap). It samples continuous vectors from this energy-based model and round them to text.

AR-infilling. We train an autoregressive LM from scratch to do sentence infilling task [9]. Similar to training Diffusion-LM, we train on the ROCStories dataset, but pre-process it by reordering sentences from $(O_1, O_{\text{middle}}, O_2)$ to $(O_1, O_2, O_{\text{middle}})$. At evaluation time, we feed in O_1, O_2, and the model generates the middle sentence.

7 Main Results

We train Diffusion-LMs on the E2E and ROCStories datasets. In terms of negative log-likelihood (NLL, lower is better), we find that the variational upper bound of Diffusion-LM NLL [10] underperforms the equivalent autoregressive Transformer model (2.28 vs. 1.77 for E2E, 3.88 vs 3.05 for ROCStories) although scaling up model and dataset size partially bridges the gap (3.88 → 3.10 on ROCStories). Our best log-likelihoods required several modifications from §4; we explain these and give detailed log-likelihood results in Appendix F. Despite worse likelihoods, controllable generation based on our Diffusion-LM results in significantly better outputs than systems based on autoregressive LMs, as we will show in §7.1,§7.2, and §7.3

7.1 Classifier-Guided Controllable Text Generation Results

As shown in Table 2, Diffusion-LM achieves high success and fluency across all classifier-guided control tasks. It significantly outperforms the PPLM and FUDGE baselines across all 5 tasks. Surprisingly, our method outperforms the fine-tuning oracle on controlling syntactic parse trees and spans, while achieving similar performance on the remaining 3 tasks.

[10] Exact log-likelihoods are intractable for Diffusion-LM, so we report the lower bound $\mathcal{L}_{\text{vlb}}^{\text{e2e}}$.

	Semantic Content		Parts-of-speech		Syntax Tree		Syntax Spans		Length	
	ctrl ↑	lm ↓	ctrl ↑	lm ↓	ctrl ↑	lm ↓	ctrl ↑	lm ↓	ctrl ↑	lm ↓
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
Diffusion-LM	**81.2**	**2.55**	**90.0**	**5.16**	**86.0**	3.71	**93.8**	**2.53**	**99.9**	**2.16**
FT-sample	72.5	2.87	89.5	4.72	64.8	5.72	26.3	2.88	98.1	3.84
FT-search	89.9	1.78	93.0	3.31	76.4	3.24	54.4	2.19	100.0	1.83

Table 2: Diffusion-LM achieves high success rate (ctrl ↑) and good fluency (lm ↓) across all 5 control tasks, outperforming the PPLM and FUDGE baselines. Our method even outperforms the fine-tuning oracle (FT) on controlling syntactic parse trees and spans.

Syntactic Parse	(S (S (NP *) (VP * (NP (NP * *) (VP * **(NP (ADJP * *) *)**))))) * (S (NP * * *) (VP * (ADJP (ADJP *)))))
FUDGE	Zizzi is a cheap restaurant . **[incomplete]**
Diffusion-LM	Zizzi is a pub providing **family friendly Indian food** Its customer rating is low
FT	Cocum is a Pub serving **moderately priced meals** and the customer rating is high
Syntactic Parse	(S (S (VP * (PP * (NP * *)))) * **(NP * * *)** (VP * (NP (NP * *) (SBAR (WHNP *) (S (VP * (NP * *)))))) *)
FUDGE	In the city near The Portland Arms is a coffee and fast food place named The Cricketers which is not family - friendly with a customer rating of 5 out of 5 .
Diffusion-LM	Located on the riverside , **The Rice Boat** is a restaurant that serves Indian food .
FT	Located near The Sorrento, **The Mill** is a pub that serves Indian cuisine.

Table 3: Qualitative examples from the Syntax Tree control. The syntactic parse tree is linearized by nested brackets representing the constituents, and we use the standard PTB syntactic categories. Tokens within each span are represented as * . We color failing spans red and **bold** the spans of interest that we discuss in §7.1.

Controlling syntactic parse trees and spans are challenging tasks for fine-tuning, because conditioning on the parse tree requires reasoning about the nested structure of the parse tree, and conditioning on spans requires lookahead planning to ensure the right constituent appears at the target position.

We observe that PPLM fails in semantic content controls and conjecture that this is because PPLM is designed to control coarse-grained attributes, and may not be useful for more targeted tasks such as enforcing that a restaurant review contains a reference to Starbucks.

FUDGE performs well on semantic content control but does not perform well on the remaining four tasks. Controlling a structured output (Parts-of-speech and Syntax Tree) is hard for FUDGE because making one mistake anywhere in the prefix makes the discriminator assign low probabilities to all continuations. In other control tasks that require planning (Length and Syntax Spans), the future discriminator is difficult to train, as it must implicitly perform lookahead planning.

The non-autoregressive nature of our Diffusion-LM allows it to easily solve all the tasks that require precise future planning (Syntax Spans and Length). We believe that it works well for complex controls that involve global structures (Parts-of-speech, Syntax Tree) because the coarse-to-fine representations allow the classifier to exert control on the entire sequence (near $t = T$) as well as on individual tokens (near $t = 0$).

Qualitative Results. Table 3 shows samples of Syntax Tree control. Our method and fine-tuning both provide fluent sentences that mostly satisfy controls, whereas FUDGE deviates from the constraints after the first few words. One key difference between our method and fine-tuning is that Diffusion-LM is able to correct for a failed span and have suffix spans match the target. In the first example, the generated span ("Family friendly Indian food") is wrong because it contains 1 more word than the target. Fortunately, this error doesn't propagate to later spans, since Diffusion-LM adjusts by dropping the conjunction. Analogously, in the second example, the FT model generates a failed span ("The Mill") that contains 1 fewer word. However, the FT model fails to adjust in the suffix, leading to many misaligned errors in the suffix.

	Semantic Content + Syntax Tree			Semantic Content + Parts-of-speech		
	semantic ctrl ↑	syntax ctrl ↑	lm ↓	semantic ctrl ↑	POS ctrl ↑	lm ↓
FUDGE	61.7	15.4	3.52	64.5	24.1	3.52
Diffusion-LM	**69.8**	**74.8**	5.92	**63.7**	**69.1**	3.46
FT-PoE	61.7	29.2	**2.77**	29.4	10.5	**2.97**

Table 4: In this experiment, we compose semantic control and syntactic control: Diffusion-LM achieves higher success rate (ctrl ↑) at some cost of fluency (lm ↓). Our method outperforms both FUDGE and FT-PoE (product of experts of two fine-tuned models) on control success rate, especially for the structured syntactic controls (i.e. syntactic parse tree and POS).

	Automatic Eval				Human Eval
	BLEU-4 ↑	ROUGE-L ↑	CIDEr ↑	BERTScore ↑	
Left-only	0.9	16.3	3.5	38.5	n/a
DELOREAN	1.6	19.1	7.9	41.7	n/a
COLD	1.8	19.5	10.7	42.7	n/a
Diffusion	**7.1**	**28.3**	**30.7**	**89.0**	$\mathbf{0.37}^{+0.03}_{-0.02}$
AR	6.7	27.0	26.9	**89.0**	$\mathbf{0.39}^{+0.02}_{-0.03}$

Table 5: For sentence infilling, Diffusion-LM significantly outperforms prior work COLD [31] and Delorean [30] (numbers taken from paper), and matches the performance of an autoregressive LM (AR) trained from scratch to do infilling.

7.2 Composition of Controls

One unique capability of plug-and-play controllable generation is its modularity. Given classifiers for multiple independent tasks, gradient guided control makes it simple to generate from the intersection of multiple controls by taking gradients on the sum of the classifier log-probabilities.

We evaluate this setting on the combination of Semantic Content + Syntax Tree control and Semantic Content + Parts-of-speech control. As shown in Table 4, our Diffusion-LM achieves a high success rate for both of the two components, whereas FUDGE gives up on the more global syntactic control. This is expected because FUDGE fails to control syntax on its own.

Fine-tuned models are good at POS and semantic content control individually but do not compose these two controls well by product of experts (PoE), leading to a large drop in success rates for both constraints.

7.3 Infilling Results

As shown in Table 5, our diffusion LM significantly outperforms continuous relaxation based methods for infilling (COLD and DELOREAN). Moreover, our method achieves comparable performance to fine-tuning a specialized model for this task. Our method has slightly better automatic evaluation scores and the human evaluation found no statistically significant improvement for either method. These results suggest that Diffusion LM can solve many types of controllable generation tasks that depend on generation order or lexical constraints (such as infilling) without specialized training.

7.4 Ablation Studies

We verify the importance of our proposed design choices in §4 through two ablation studies. We measure the sample quality of Diffusion-LM using the lm-score on 500 samples §6.2.

Learned v.s. Random Embeddings (§4.1). Learned embeddings outperform random embeddings on the ROCStories, which is a harder language modeling task. The same trend holds for the E2E dataset but with a smaller margin.

Objective Parametrization (§4.2). We propose to let the diffusion model predict $\mathbf{x}_0$ directly. Here, we compare this with standard parametrization in image generation which parametrizes by the noise term ϵ. Figure 4 (right) shows that parametrizing by $\mathbf{x}_0$ consistently attains good performance across dimensions, whereas parametrizing by ϵ works fine for small dimensions, but quickly collapses for larger dimensions.

8 Conclusion and Limitations

We proposed Diffusion-LM, a novel and controllable language model based on continuous diffusions, which enables new forms of complex fine-grained control tasks. We demonstrate Diffusion-LM's success in 6 fine-grained control tasks: our method almost doubles the control success rate of prior methods and is competitive with baseline fine-tuning methods that require additional training.

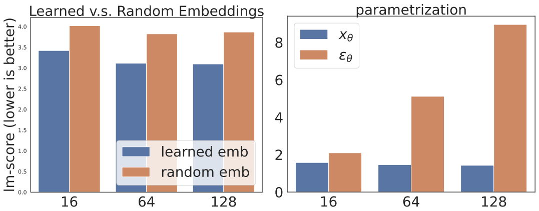

Figure 4: We measure the impact of our proposed design choices through lm-score. We find both learned embeddings and reparametrization substantially improves sample quality.

We find the complex controls enabled by Diffusion-LM to be compelling, and we are excited by how Diffusion-LM is a substantial departure from the current paradigm of discrete autoregressive generation. As with any new technologies, there are drawbacks to the Diffusion-LMs that we constructed: (1) it has higher perplexity; (2) decoding is substantially slower; and (3) training converges more slowly. We believe that with more follow-up work and optimization, many of these issues can be addressed, and this approach will turn out to be a compelling way to do controllable generation at scale.

Acknowledgments and Disclosure of Funding

We thank Yang Song, Jason Eisner, Tianyi Zhang, Rohan Taori, Xuechen Li, Niladri Chatterji, and the members of p-lambda group for early discussions and feedbacks. We gratefully acknowledge the support of a PECASE award. Xiang Lisa Li is supported by a Stanford Graduate Fellowship.

References

[1] Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=h7-XixPCAL.

[2] Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Wen tau Yih, and Yejin Choi. Abductive commonsense reasoning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=Byg1v1HKDB.

[3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[4] Nanxin Chen, Yu Zhang, Heiga Zen, Ron J Weiss, Mohammad Norouzi, and William Chan. Wavegrad: Estimating gradients for waveform generation. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=NsMLjcFaO8O.

[5] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh,

Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam M. Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Benton C. Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier García, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Oliveira Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathleen S. Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

[6] Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=H1edEyBKDS`.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805, 2019.

[8] Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat GANs on image synthesis. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL `https://openreview.net/forum?id=AAWuCvzaVt`.

[9] Chris Donahue, Mina Lee, and Percy Liang. Enabling language models to fill in the blanks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2492–2501, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.225. URL `https://aclanthology.org/2020.acl-main.225`.

[10] John C. Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. In *J. Mach. Learn. Res.*, 2010.

[11] Jiatao Gu, James Bradbury, Caiming Xiong, Victor O.K. Li, and Richard Socher. Non-autoregressive neural machine translation. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=B1l8BtlCb`.

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, pages 6840–6851, 2020.

[13] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=rygGQyrFvH`.

[14] Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Towards non-autoregressive language models. *arXiv preprint arXiv:2102.05379*, 2021.

[15] Emiel Hoogeboom, Alexey A. Gritsenko, Jasmijn Bastings, Ben Poole, Rianne van den Berg, and Tim Salimans. Autoregressive diffusion models. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=Lm8T39vLDTE`.

[16] N. Keskar, B. McCann, L. R. Varshney, Caiming Xiong, and R. Socher. Ctrl: A conditional transformer language model for controllable generation. *ArXiv*, abs/1909.05858, 2019.

[17] Daniel Khashabi, Gabriel Stanovsky, Jonathan Bragg, Nicholas Lourie, Jungo Kasai, Yejin Choi, Noah A. Smith, and Daniel S. Weld. Genie: A leaderboard for human-in-the-loop evaluation of text generation. *ArXiv*, abs/2101.06561, 2021.

[18] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *International Conference on Learning Representations (ICLR)*, 2014.

[19] Diederik P Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *arXiv preprint arXiv:2107.00630*, 2021.

[20] Nikita Kitaev and Dan Klein. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1249. URL `https://aclanthology.org/P18-1249`.

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

[22] Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. GeDi: Generative Discriminator Guided Sequence Generation. *arXiv preprint arXiv:2009.06367*, 2020.

[23] Shankar Kumar and William Byrne. Minimum Bayes-risk decoding for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 169–176, Boston, Massachusetts, USA, May 2 - May 7 2004. Association for Computational Linguistics. URL `https://aclanthology.org/N04-1022`.

[24] Alisa Liu, Maarten Sap, Ximing Lu, Swabha Swayamdipta, Chandra Bhagavatula, Noah A. Smith, and Yejin Choi. DExperts: Decoding-time controlled text generation with experts and anti-experts. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6691–6706, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.522. URL `https://aclanthology.org/2021.acl-long.522`.

[25] Gautam Mittal, Jesse Engel, Curtis Hawthorne, and Ian Simon. Symbolic music generation with diffusion models. *arXiv preprint arXiv:2103.16091*, March 2021.

[26] 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, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1098. URL `https://aclanthology.org/N16-1098`.

[27] Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. *arXiv preprint arXiv:2102.09672*, 2021.

[28] Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. The E2E dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206, Saarbrücken, Germany, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-5525. URL `https://aclanthology.org/W17-5525`.

[29] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL `https://aclanthology.org/D14-1162`.

[30] Lianhui Qin, Vered Shwartz, Peter West, Chandra Bhagavatula, Jena D. Hwang, Ronan Le Bras, Antoine Bosselut, and Yejin Choi. Back to the future: Unsupervised backprop-based decoding for counterfactual and abductive commonsense reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 794–805, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.58. URL `https://aclanthology.org/2020.emnlp-main.58`.

[31] Lianhui Qin, Sean Welleck, Daniel Khashabi, and Yejin Choi. Cold decoding: Energy-based constrained text generation with langevin dynamics, 2022. URL `https://arxiv.org/abs/2202.11705`.

[32] Alec Radford and Karthik Narasimhan. Improving language understanding by generative pre-training. `https://openai.com/blog/language-unsupervised/`, 2018.

[33] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. https://openai.com/blog/better-language-models/, 2019.

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

[35] Yi Ren, Jinglin Liu, Xu Tan, Zhou Zhao, Sheng Zhao, and Tie-Yan Liu. A study of non-autoregressive model for sequence generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 149–159, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.15. URL `https://aclanthology.org/2020.acl-main.15`.

[36] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

[37] Chitwan Saharia, William Chan, Saurabh Saxena, and Mohammad Norouzi. Non-autoregressive machine translation with latent alignments. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1098–1108, 2020.

[38] Lei Sha. Gradient-guided unsupervised lexically constrained text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8692–8703, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.701. URL `https://aclanthology.org/2020.emnlp-main.701`.

[39] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2256–2265, Lille, France, 07–09 Jul 2015. PMLR. URL `https://proceedings.mlr.press/v37/sohl-dickstein15.html`.

[40] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=PxTIG12RRHS`.

[41] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 2008.

[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf`.

[43] Ben Wang and Aran Komatsuzaki. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`, May 2021.

[44] Kevin Yang and Dan Klein. FUDGE: Controlled text generation with future discriminators. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3511–3535, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.276. URL `https://aclanthology.org/2021.naacl-main.276`.

[45] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. Opt: Open pre-trained transformer language models, 2022. URL `https://arxiv.org/abs/2205.01068`.

A Diffusion Noise Schedule

Because a diffusion model shares parameters for all diffusion steps, the noise schedule (parametrized by $\bar{\alpha}_{1:T}$) is an important hyperparameter that determines how much weight we assign to each denoising problem. We find that standard noise schedules for continuous diffusions are not robust for text data. We hypothesize that the discrete nature of text and the rounding step make the model insensitive to noise near $t = 0$. Concretely, adding small amount of Gaussian noise to a word embedding is unlikely to change its nearest neighbor in the embedding space, making denoising an easy task near $t = 0$.

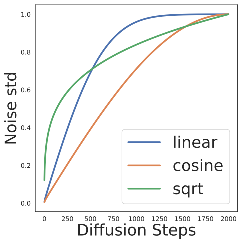

Figure 5: Visualizing the noise schedule $\sqrt{1 - \bar{\alpha}_t}$.

To address this, we introduce a new *sqrt* noise schedule that is better suited for text, shown in Figure 5 defined by $\bar{\alpha}_t = 1 - \sqrt{t/T + s}$, where s is a small constant that corresponds to the starting noise level[11]. Compared to standard *linear* and *cosine* schedules, our *sqrt* schedule starts with a higher noise level and increase noise rapidly for the first 50 steps. Then *sqrt* slows down injecting noise to avoid spending much steps in the high-noise problems, which may be too difficult to solve well.

B Hyperparameters

Diffusion-LM hyperparameters. The hyperparameters that are specific to Diffusion-LM include the number of diffusion steps, the architecture of the Diffusion-LM, the embedding dimension, and the noise schedule, . We set the diffusion steps to be 2000, the architecture to be BERT-base [7], and the sequence length to be 64. For the embedding dimensions, we select from $d \in \{16, 64, 128, 256\}$ and select $d = 16$ for the E2E dataset and $d = 128$ for ROCStories. For the noise schedule, we design the *sqrt* schedule (Appendix A) that is more robust to different parametrizations and embedding dimensions as shown in Appendix H. However, once we picked the $\mathbf{x}_0$-parametrization (§4.2) the advantage of *sqrt* schedule is not salient.

Training hyperparameters. We train Diffusion-LMs using AdamW optimizer and a linearly decay learning rate starting at 1e-4, dropout of 0.1, batch size of 64, and the total number of training iteration is 200K for E2E dataset, and 800K for ROCStories dataset. Our Diffusion-LMs are trained on a single GPU: NVIDIA RTX A5000, NVIDIA GeForce RTX 3090, or NVIDIA A100. It takes approximately 5 hours to train for 200K iterations on a single A100 GPU.

To stablize the training under $\mathcal{L}^{\text{e2e}}_{\text{vlb}}$ objective, we find that we need to set gradient clipping to 1.0 and apply importance sampling to reweight each term in $\mathcal{L}_{\text{vlb}}$ [27]. Both tricks are not necessary for $\mathcal{L}^{\text{e2e}}_{\text{simple}}$ objective.

Controllable Generation hyperparameters. To achieve controllable generation, we run gradient update on the continuous latents of Diffusion-LM. We use the AdaGrad optimizer [10] to update the latent variables, and we tune the learning rate, $\text{lr} \in \{0.05, 0.1, 0.15, 0.2\}$ and the trade-off parameter $\lambda \in \{0.1, 0.01, 0.001, 0.0005\}$. Different plug-and-play controllable generation approaches tradeoff between fluency and control by tunning different hyperparameters: PPLM uses the number of gradient updates per token, denoted as k, and we tune $k \in \{10, 30\}$. FUDGE uses the tradeoff parameter λ_{FUDGE} and we tune this $\lambda_{\text{FUDGE}} \in \{16, 8, 4, 2\}$. Table 6 contains all the selected hyperparameter for each control tasks. Both PPLM and FUDGE has additional hyperparameters and we follow the instruction from the original paper to set those. For PPLM, we set the learning rate to be 0.04 and KL-scale to be 0.01. For FUDGE, we set precondition top-K to be 200, post top-K to be 10.

C Decoding Speed

Sampling from Diffusion-LMs requires iterating through the 2000 diffusion steps, yielding $O(2000)$ f_θ model calls. In contrast, sampling from autoregressive LMs takes $O(n)$ where n is the sequence length. Therefore, decoding Diffusion-LM is slower than decoding autoregressive LMs in short and

[11] We set $s =$1e-4, and $T = 2000$, which sets the initial standard deviation to 0.1.

	Semantic		Parts-of-speech		Syntax Tree		Syntax Spans		Length	
	tradeoff	lr	tradeoff	lr	tradeoff	lr	tradeoff	lr	tradeoff	lr
PPLM	30	0.04	-	-	-	-	-	-	-	-
FUDGE	8.0	-	20.0	-	20.0	-	20.0	-	2.0	-
Diffusion-LM	0.01	0.1	0.0005	0.05	0.0005	0.2	0.1	0.15	0.01	0.1

Table 6: Hyperparameters for controllable generation methods.

medium-length sequence regimes. Concretely, it takes around 1 minute to decode 50 sequence of length 64.

To speed up decoding, we tried skipping steps in the generative diffusion process and downsample 2000 steps to 200 steps. Concretely, we set $T = 200$ and downsample the noise schedule $\bar{\alpha}_t = \bar{\alpha}_{10t}$, which is equivalent to setting each unit transition as the transition $\mathbf{x}_t \to \mathbf{x}_{t+10}$. We decode Diffusion-LM using this new noise schedule and discretization. We find that this naive approach doesn't hurt sample quality for simple language modeling tasks like E2E, but it hurts sample quality for harder language modeling tasks like ROCStories.

For plug-and-play controllable generation tasks, extant approaches are even slower. PPLM takes around 80 minutes to generate 50 samples (without batching), because it needs to run 30 gradient updates for each token. FUDGE takes 50 seconds to generate 50 samples (with batching), because it needs to call the lightweight classifier for each partial sequence, requiring 200 classifier calls for each token, yielding $100\times$ sequence length calls. We can batch the classifier calls, but it sometimes limits batching across samples due to limited GPU memory. Our Diffusion-LM takes around 80 seconds to generate 50 samples (with batching). Our method downsamples the number of diffusion steps to 200, and it takes 3 classifier calls per diffusion step, yielding 600 model calls in total.

D Classifiers for Classifier-Guided Controls

Semantic Content. We train an autoregressive LM (GPT-2 small architecture) to predict the (field, value) pair conditioned on text. To parametrize $\log p(\mathbf{c} \mid \mathbf{x}_t)$, we compute the logprob of "value" per token.

Parts-of-speech. The classifier is parametrized by a parts-of-speech tagger, which estimates the probability of the target POS sequence conditioned on the latent variables. This tagger uses a BERT-base architecture: the input is the concantenated word embedding, and output a softmax distribution over all POS tags for each input word. $\log p(\mathbf{c} \mid \mathbf{x}_t)$ is the sum of POS log-probs for each word in the sequence.

Syntax Tree. We train a Transformer-based constituency parser [20]. Our parser makes locally normalized prediction for each span, predicting either "not a constituent", or a label for the constituent (e.g., Noun Phrase). $\log p(\mathbf{c} \mid \mathbf{x}_t)$ is the sum of log-probs for each labeled and non-constituency span in the sequence.

Syntax Span. We use the same parser trained for the syntax tree. $\log p(\mathbf{c} \mid \mathbf{x}_t)$ is the log-probability that the target span is annotated with the target label.

E End-to-end Objective Derivations

For continuous diffusion models (§3.3), $\mathcal{L}_{\text{simple}}$ is derived from the canonical objective $\mathcal{L}_{\text{vlb}}$ by reweighting each term. The first T terms in $\mathcal{L}_{\text{vlb}}$ are all KL divergence between two Gaussian distributions, which has a closed form solution. Take the t-th term for example:

$$\mathop{\mathbb{E}}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[\log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_0, \mathbf{x}_t)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}\right] = \mathop{\mathbb{E}}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[\frac{1}{2\sigma_t^2}||\mu_\theta(\mathbf{x}_t, t) - \hat{\mu}(\mathbf{x}_t, \mathbf{x}_0)||^2\right] + C, \tag{3}$$

where C is a constant, $\hat{\mu}$ is the mean of the posterior $q(\mathbf{x}_{t-1}|\mathbf{x}_0, \mathbf{x}_t)$, and μ_θ is the mean of $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$ predicted by the diffusion model. Intuitively, this simplification matches the predicted mean of $\mathbf{x}_{t-1}$ to its true posterior mean. The simplification involves removing the constant C and the scaling factor $\frac{1}{2\sigma_t^2}$, yielding one term in $\mathcal{L}_{\text{simple}}$: $\mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[||\mu_\theta(\mathbf{x}_t, t) - \hat{\mu}(\mathbf{x}_t, \mathbf{x}_0)||^2\right]$.

To apply continuous diffusion to model discrete text, we design Diffusion-LM (§4.1) and propose a new end-to-end training objective (equation (2)) that learns the diffusion model and the embedding parameters jointly. The $\mathcal{L}^{\text{e2e}}_{\text{vlb}}$ can be written out as

$$\begin{aligned}
\mathcal{L}^{\text{e2e}}_{\text{vlb}}(\mathbf{w}) &= \mathop{\mathbb{E}}_{q_\phi(\mathbf{x}_0|\mathbf{w})}\left[\mathcal{L}_{\text{vlb}}(\mathbf{x}_0) + \log q_\phi(\mathbf{x}_0|\mathbf{w}) - \log p_\theta(\mathbf{w}|\mathbf{x}_0)]\right] \\
&= \mathop{\mathbb{E}}_{q_\phi(\mathbf{x}_{0:T}|\mathbf{w})}\left[\underbrace{\log \frac{q(\mathbf{x}_T|\mathbf{x}_0)}{p_\theta(\mathbf{x}_T)}}_{L_T} + \sum_{t=2}^{T}\underbrace{\log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_0,\mathbf{x}_t)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}}_{L_{t-1}} - \underbrace{\frac{\log q_\phi(\mathbf{x}_0|\mathbf{w})}{\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}}_{L_0} - \underbrace{\log p_\theta(\mathbf{w}|\mathbf{x}_0)}_{L_{\text{round}}}\right]
\end{aligned}$$

We apply the same simplification which transforms $\mathcal{L}_{\text{vlb}} \to \mathcal{L}_{\text{simple}}$ to transform $\mathcal{L}^{\text{e2e}}_{\text{vlb}} \to \mathcal{L}^{\text{e2e}}_{\text{simple}}$:

$$\begin{aligned}
\mathop{\mathbb{E}}_{q_\phi(\mathbf{x}_{0:T}|\mathbf{w})}[L_T] &\to \mathbb{E}[||\mathop{\mathbb{E}}_{\mathbf{x}_T\sim q}[\mathbf{x}_T|\mathbf{x}_0] - 0||^2] = \mathbb{E}[||\hat{\mu}(\mathbf{x}_T;\mathbf{x}_0)]||^2] \\
\mathop{\mathbb{E}}_{q_\phi(\mathbf{x}_{0:T}|\mathbf{w})}[L_{t-1}] &\to \mathbb{E}[||\mathop{\mathbb{E}}_{\mathbf{x}_{t-1}\sim q}[\mathbf{x}_{t-1}|\mathbf{x}_0,\mathbf{x}_t] - \mathop{\mathbb{E}}_{\mathbf{x}_{t-1}\sim p_\theta}[\mathbf{x}_{t-1}|\mathbf{x}_t]||^2] = \mathbb{E}[||\hat{\mu}(\mathbf{x}_t,\mathbf{x}_0) - \mu_\theta(\mathbf{x}_t,t)||^2] \\
\mathop{\mathbb{E}}_{q_\phi(\mathbf{x}_{0:T}|\mathbf{w})}[L_0] &\to \mathbb{E}[||\mathop{\mathbb{E}}_{\mathbf{x}_0\sim q_\phi}[\mathbf{x}_0 \mid \mathbf{w}] - \mathop{\mathbb{E}}_{\mathbf{x}_0\sim p_\theta}[\mathbf{x}_0 \mid \mathbf{x}_1]||^2] = \mathbb{E}[||\textsc{Emb}(w) - \mu_\theta(\mathbf{x}_1,1)||^2]
\end{aligned}$$

It's worth noting that the first term is constant if the noise schedule satisfies $\bar{\alpha}_T = 0$, which guarantees $\mathbf{x}_T$ is pure Gaussian noise. In contrast, if the noise schedule doesn't go all the way such that $\mathbf{x}_T$ is pure Gaussian noise, we need to include this regularization term to prevent the embedding from learning too large norms. Embedding with large norms is a degenerate solution, because it is impossible to sample from $p(\mathbf{x}_T)$ accurately, even though it makes all the other denoising transitions easily predictable.

Combining these terms yield $\mathcal{L}^{\text{e2e}}_{\text{simple}}$.

$$\begin{aligned}
\mathcal{L}^{\text{e2e}}_{\text{simple}}(\mathbf{w}) = \mathop{\mathbb{E}}_{q_\phi(\mathbf{x}_{0:T}|\mathbf{w})}&\left[||\hat{\mu}(\mathbf{x}_T;\mathbf{x}_0)||^2 + \sum_{t=2}^{T}[||\hat{\mu}(\mathbf{x}_t,\mathbf{x}_0) - \mu_\theta(\mathbf{x}_t,t)||^2]\right] \\
&+ \mathop{\mathbb{E}}_{q_\phi(\mathbf{x}_{0:1}|\mathbf{w})}\left[||\textsc{Emb}(\mathbf{w}) - \mu_\theta(\mathbf{x}_1,1)||^2 - \log p_\theta(\mathbf{w}|\mathbf{x}_0)\right].
\end{aligned}$$

Intuitively, we learn a Transformer model that that takes as input $(\mathbf{x}_t, t) \in (\mathbb{R}^{nd}, \mathbb{R})$ and the goal is to predict the distribution of $\mathbf{x}_{t-1} \in \mathbb{R}^{nd}$. It's worth noting that this Transformer model is shared across all the diffusion steps $t = 1 \ldots T$. As we demonstrated in the derivation of $\mathcal{L}^{\text{e2e}}_{\text{simple}}$, the most natural thing is to directly parametrize the neural network to predict the mean of $\mathbf{x}_{t-1} \mid \mathbf{x}_t$, we call this μ_θ-parametrization.

There are other parametrizations that are equivalent to μ_θ-parametrization up to a scaling constant. For example in §4.2, we can train the Transformer model to directly predict $\mathbf{x}_0$ via $f_\theta(\mathbf{x}_t, t)$, and use the tractable Gaussian posterior $q(\mathbf{x}_{t-1} \mid \mathbf{x}_0, \mathbf{x}_t)$ to compute the mean of $\mathbf{x}_{t-1}$, which has a closed form solution, conditioned on predicted $\mathbf{x}_0$ and observed $\mathbf{x}_t$: $\frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t$.

$$\begin{aligned}
&||\hat{\mu}(\mathbf{x}_t,\mathbf{x}_0) - \mu_\theta(\mathbf{x}_t,t)||^2 \\
=&||(\frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t) - (\frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}f_\theta(\mathbf{x}_t,t) + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t)||^2 \\
=&||\frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}(\mathbf{x}_0 - f_\theta(\mathbf{x}_t,t))||^2 \\
\propto&||\mathbf{x}_0 - f_\theta(\mathbf{x}_t,t)||^2
\end{aligned}$$

Dataset	Small AR	Small Diffusion	Medium Diffusion
E2E	1.77	2.28	-
ROCStories	3.05	3.88	-
ROCStories (+GPT-J)	2.41	3.59	3.10

Table 7: Log-likelihood results (nats per token)

These two parametrizations differ by a constant scaling, and we apply the $\mathbf{x}_0$-parametrization to all terms in $\mathcal{L}^{\text{e2e}}_{\text{simple}}$ to reduce rounding errors as discussed in §4.2:

$$\mathcal{L}^{\text{e2e}}_{\mathbf{x}_0\text{-simple}}(\mathbf{w}) = \mathop{\mathbb{E}}_{q_\phi(\mathbf{x}_{0:T}|\mathbf{w})}\left[||\hat{\mu}(\mathbf{x}_T;\mathbf{x}_0)||^2 + \sum_{t=2}^{T}[||\mathbf{x}_0 - f_\theta(\mathbf{x}_t, t)||^2]\right] + \mathop{\mathbb{E}}_{q_\phi(\mathbf{x}_{0:1}|\mathbf{w})}\left[||\textsc{Emb}(\mathbf{w}) - f_\theta(\mathbf{x}_1, 1)||^2 - \log p_\theta(\mathbf{w}|\mathbf{x}_0)\right].$$

To generate samples from a Diffusion-LM with $\mathbf{x}_0$-parametrization, at each diffusion step, the model estimates the $\mathbf{x}_0$ via $f_\theta(\mathbf{x}_t, t)$ and then we sample $\mathbf{x}_{t-1}$ from $q(\mathbf{x}_{t-1} \mid f_\theta(\mathbf{x}_t, t), \mathbf{x}_t)$, which is fed as input to the next diffusion step.

F Log-Likelihood Models and Results

To investigate Diffusion-LM's log-likelihood performance, we make several departures from the training procedure of §4. Ultimately the log-likelihood improvements described in this section did not translate into better generation quality in our experiments and therefore we focus on the original method in the rest of the paper. Our likelihood models are trained as follows:

- Instead of training a diffusion model on sequences of low-dimensional token embeddings, we train a model directly sequences of on one-hot token vectors.
- Following the setup of Kingma et al. [19], we train a continuous-time diffusion model against the log-likelihood bound and learn the noise schedule simultaneously with the rest of the model to minimize the loss variance.
- Because our model predicts sequences of one-hot vectors, we use a softmax nonlinearity at its output and replace all squared-error terms in the loss function with cross-entropy terms. This choice of surrogate loss led to better optimization, even though we evaluate against the original loss with squared-error terms.
- The model applies the following transformation to its inputs before any Transformer layers: $x := \operatorname{softmax}(\alpha(t)x + \beta(t))$ where $\alpha(t) \in \mathbb{R}$ and $\beta(t) \in \mathbb{R}^v$ are learned functions of the diffusion timestep t parameterized by MLPs (v is the vocabulary size).
- At inference time, we omit the rounding procedure in §4.2.

For exact model architecture and training hyperparameter details, please refer to our released code.

We train these diffusion models, as well as baseline autoregressive Transformers, on E2E and ROCStories and report log-likelihoods in Table 7. We train two sizes of Transformers: "small" models with roughly 100M parameters and "medium" models with roughly 300M parameters. Both E2E and ROCstories are small enough datasets that all of our models reach their minimum test loss early in training (and overfit after that). To additionally compare model performance in a large-dataset regime, we also present "ROCStories (+GPT-J)" experiments in which we generate 8M examples of synthetic ROCStories training data by finetuning GPT-J [43] on the original ROCStories data, pretrain our models on the synthetic dataset, and then finetune and evaluate them on the original ROCStories data.

G Qualitative Examples

We show randomly sampled outputs of Diffusion-LM both for unconditional generation and for the 5 control tasks. Table 8 shows the unconditional generation results. Table 9, Table 10, Table 12, and

Table 3 show the qualitative samples from span control, POS control, semantic content control, and syntax tree control, respectively. Table 11 shows the results of length control.

H Additional Ablation Studies

In addition to the 2 ablation studies in §7.4, we provide more ablation results in Figure 6 about architecture choices and noise schedule.

Learned v.s. Random Embeddings (§4.1). Learned embeddings outperform random embeddings on both ROCStories and the E2E dataset by xx percent and xx percent respectively, as shown in the first row of Figure 6.

Noise Schedule (Appendix A). We compare the *sqrt* schedule with *cosine* [27] and *linear* [12] schedules proposed for image modeling. The middle row of Figure 6 demonstrates that *sqrt* schedule attains consistently good and stable performance across all dimension and parametrization choices. While the *sqrt* schedule is less important with $\mathbf{x}_0$-parametrization, we see that it provides a substantially more robust noise schedule under alternative parametrizations such as ϵ.

Transformer v.s. U-Net. The U-Net architecture in Ho et al. [12] utilizes 2D-convolutional layers, and we imitate all the model architectures except changing 2D-conv to 1D-conv which is suitable for text data. Figure 6 (last row) shows that the Transformer architecture outperforms U-Net.

I Societal Impacts

On the one hand, having strong controllability in language models will help with mitigating toxicity, making the language models more reliable to deploy. Additionally, we can also control the model to be more truthful, reducing the inaccurate information generated by the language model by carefully controlling it to be truthful. On the other hand, however, one could also imagine more powerful targeted disinformation (e.g., narrative wedging) derived from the fine-grained controllability.

Towards this end, it might be worth considering generation methods that can watermark the generated outputs without affecting its fluency, and this type of watermark could also be framed as a controllable generation problem, with distinguish-ability and fluency as the constraints.

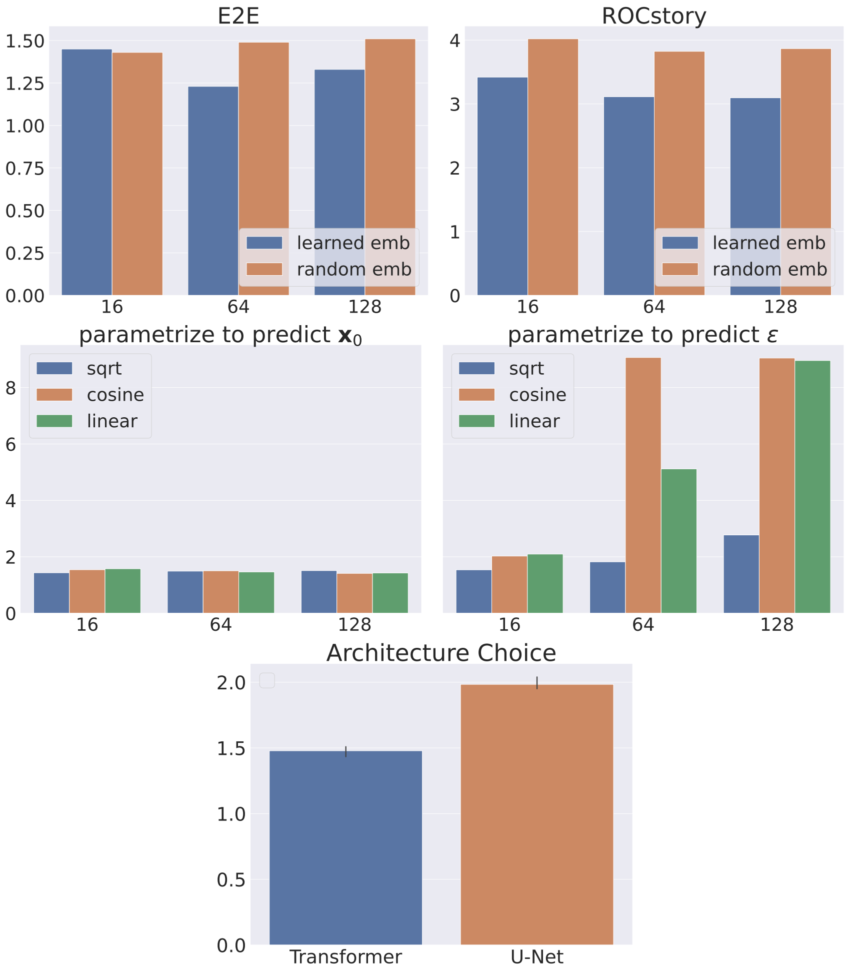

Figure 6: Additional ablation results. The first row shows Diffusion-LM with trainable embeddings outperform random embeddings on both datasets (§4.1). The second row demonstrates that *sqrt* schedule attains consistently good and stable performance across all dimension and parametrization choices. The last row shows that Transformer architecture outperforms U-Net architecture for language modeling.

ROCStories+Aug	Matt was at the store . He was looking at a new toothbrush . He found the perfect one . When he got home , he bought it . It was bright and he loved it .
	I and my friend were hungry . We were looking for some to eat . We went to the grocery store . We bought some snacks . We decided to pick up some snacks .
	I was at the store . I had no money to buy milk . I decided to use the restroom . I went to the register . I was late to work .
	The man wanted to lose weight . He did n't know how to do weight . He decided to start walking . He ate healthy and ate less . He lost ten pounds in three months .
	I went to the aquarium . I wanted to feed something . I ordered a fish . When it arrived I had to find something . I was disappointed .
ROCStories	Tom was planning a trip to California . He had fun in the new apartment . He was driving , until it began to rain . Unfortunately , he was soaked . Tom stayed in the rain at the beach .
	Carrie wanted a new dress . She did not have enough money . She went to the bank to get one , but saw the missed . Finally , she decided to call her mom . She could not wait to see her new dress .
	Tina went to her first football game . She was excited about it . When she got into the car she realized she forgot her hand . She ended up getting too late . Tina had to start crying .
	Michael was at the park . Suddenly he found a stray cat . He decided to keep the cat . He went to his parents and demanded a leg . His parents gave him medicine to get it safe .
	Tim was eating out with friends . They were out of service . Tim decided to have a pizza sandwich . Tim searched for several hours . He was able to find it within minutes .
E2E	The Waterman is an expensive pub that serves Japanese food . It is located in Riverside and has a low customer rating .
	A high priced pub in the city centre is The Olive Grove . It is a family friendly pub serving French food .
	The Rice Boat offers moderate priced Chinese food with a customer rating of 3 out of 5 . It is near Express by Holiday Inn .
	There is a fast food restaurant , The Phoenix , in the city centre . It has a price range of more than 0̆0a3 30 and the customer ratings are low .
	The Mill is a coffee shop based in the city centre area near to The Sorrento . It is in the high price range and serves Indian food .

Table 8: Randomly sampled examples generated by unconditional sampling Diffusion-LM trained on 3 datasets. ROCStories+Aug denotes ROCStories with data augmentation. It's generated by first fine-tuning GPT-j on the ROCStories dataset and then sample the large GPT-j model to generate 1M stories.

target span	[3, 5, PP]
FUDGE	UNK the UNK for Italian food , The Eagle coffee shop is near Burger King in the riverside area . The Eagle has a customer rating of 5 out of 5 , and isn ' t family - friendly . The Eagle has a cheap price range .
Diffusion-LM	The Plough , near Café Rouge , is a high priced fast food pub .
FT	Along the riverside near Café Rouge is The Golden Curry . It serves Italian food in a family - friendly environment . It has a low customer rating .
target span	[10, 12, PP]
FUDGE	Blue Spice is a high price range Fast food located in city centre .
Diffusion-LM	The Phoenix is a high priced food restaurant , located near the river .
FT	The Punter is a family restaurant with low prices and delicious sushi , located near the Café Sicilia
target span	[9, 14, S]
FUDGE	Zizzi pub serves Italian food for adults only . It has been rated average by customers .
Diffusion-LM	There is a Chinese restaurant called The Eagle , it has an average customer rating .
FT	On the riverside area are located Alimentum , has a very good French food for adults and kids , UNK price range are over 20 to 25 £ .
target span	[4, 16, VP]
FUDGE	The Cambridge Blue pub is near the Café Brazil and offers a high price range for their French food .
Diffusion-LM	On the Ranch there is a children friendly pub called The Cricketers with an average customer rating .
FT	The Travellers Rest Beefeater is an average rated restaurant located in the riverside area near Café Adriatic . Their price range is less than £ 20 .
target span	[0, 2, NP]
FUDGE	The Golden Palace is a cheap , 5 - star coffee shop , located on the river in the north of the city centre .
Diffusion-LM	The Olive Grove is a pub that provides Indian food in the high price range . It is in the city centre .
FT	The Golden Curry is located in city centre near Café Rouge which provides English food . Its customer rating is average and is not family - friendly .
target span	[12, 13, NP]
FUDGE	The Waterman is a family friendly place with a good rating .[missing span]
Diffusion-LM	The Vaults is a high priced , family friendly restaurant that serves Italian food .
FT	Strada is a restaurant which costs less than £ 20 , but is not family - friendly and has an average rating .

Table 9: Qualitative output of the syntax span control tasks. The target span (i, j,label) means the span from position i to position j should be a constituent with a specific label: S is sentence, NP is noun phrase, VP is verb phrase, PP is prepositional phrase, etc. We color failed spans red and correct spans green.

target POS	PROPN AUX DET ADJ NOUN NOUN VERB ADP DET NOUN ADP DET NOUN PUNCT
FUDGE	Aromi is a non family - friendly fast food coffee shop in the riverside area with a low Customer Rating .
Diffusion-LM	Fitzbillies is a cheap coffee shop located on the outskirts of the city .
FT	Aromi is a fast food pub located at the centre of the city.
target POS	PROPN AUX DET NOUN VERB NOUN ADJ NOUN PUNCT PRON NOUN NOUN AUX ADJ
FUDGE	Cocum is a family - friendly coffee shop , that has a low price range and a low customer rating .
Diffusion-LM	Zizzi is a pub providing restaurant Chinese food . Its customer rating is low
FT	Zizzi is a pub providing kids friendly services. Its customer rating is average
target POS	DET NOUN PUNCT PROPN VERB ADJ CCONJ ADJ NOUN CCONJ AUX VERB ADP DET PROPN ADJ PROPN PUNCT
FUDGE	A child - friendly coffee shop , Cocum , offers fast food at an average price range of £ 20 - 25 .
Diffusion-LM	The Waterman - friendly serves UNK and fast food and is located near the Crown Plaza Hotel .
FT	The wine - Strada serves fast and cheap food and is located near the Rainbow Vegetarian Café.
target POS	DET PROPN PROPN VERB ADJ NOUN ADP NOUN ADP SYM NUM PUNCT NOUN NOUN AUX ADJ PUNCT DET PROPN PROPN AUX VERB ADP DET PROPN CCONJ PROPN ADP PROPN PROPN PUNCT ADJ PUNCT DET NOUN PART AUX VERB PUNCT
FUDGE	The Midsummer House offers cheap English food near All Bar One . Rated 5 out of 5 .
Diffusion-LM	The Rice Boat provides Chinese food in £ 20 - 25 . Price range is high . The Rice Boat is located near the Express by Holiday Inn and is kids friendly . The customer rating is high .
FT	The Rice Boat welcomes Japanese food with prices under £ 20. Customer ratings are low. The Rice Boat is located near the Express by Holiday Inn. Convenient. No children's are allowed.
target POS	PROPN PROPN AUX DET ADJ NOUN NOUN ADP DET NOUN NOUN ADP DET PROPN PUNCT PRON AUX NOUN PUNCT ADJ PUNCT
FUDGE	Loch Fyne is a Japanese restaurant with a moderate price range and kid - friendly atmosphere .
Diffusion-LM	Browns Cambridge is an Italian restaurant shop in the city centre near The Sorrento . It is family - friendly .
FT	Browns Cambridge is a cheap coffee shop in the riverside area near The Sorrento, that is family - friendly.
target POS	PROPN VERB DET ADJ NOUN NOUN PROPN PUNCT PRON AUX ADJ VERB CCONJ VERB NOUN SCONJ VERB ADJ NOUN PUNCT
FUDGE	Fitzbillies coffee shop has a high price range , children friendly service and serves Japanese food in riverside with high customer rating .
Diffusion-LM	There has a high customer rating . It is kid friendly called The Golden Curry and serves Indian food .
FT	Customers give the French coffee shop Fitzbillies ; it is average rated and offers families where serving light meals.
target POS	DET NUM NUM VERB ADJ NOUN
FUDGE	The Twenty Two serves Fast food and is kids friendly .
Diffusion-LM	The Twenty Two provides Chinese food
FT	The Twenty Two provides Indian food
target POS	ADV NOUN ADV ADP PROPN PROPN PUNCT DET PROPN NOUN NOUN VERB ADJ NOUN NOUN CCONJ AUX PART VERB NOUN NOUN PUNCT
FUDGE	UNK your whole family to The Wrestlers , the best UNK the UNK UNK at the river
Diffusion-LM	Located in riverside near The Sorrento , Browns Cambridge coffee shop serves Japanese food , and is not family - friendly .
FT	Even adults only at Loch Fyne, The Rice Boat coffee shop has moderate price range and does not cater kids age.
target POS	DET PROPN AUX DET NUM NOUN NOUN NOUN VERB ADP DET PROPN PROPN PUNCT
FUDGE	The Eagle is a 3 star coffee shop located near Burger King , north of the City centre that provides low - cost fast food .
Diffusion-LM	The Cricketers is a five star coffee shop located near The Portland Arms .
FT	The Vaults is a one star coffee shop located near the Café Brazil.

Table 10: Qualitative output of the POS control tasks. The target POS is the sequence of gold parts-of-speech tags the generated texts should match.

target length	7
FUDGE	Wildwood is a cheap Japanese pub . Low rating .
Diffusion-LM	The Twenty Two serves Indian food .
FT	The Mill is an Indian restaurant .
target length	12
FUDGE	The Phoenix is an average Japanese restaurant that is in the City Centre .
Diffusion-LM	The Twenty Two serves Chinese food and is not family friendly .
FT	Green Man is an average priced restaurant located near All Bar One
target length	17
FUDGE	Fitzbillies is an expensive Italian coffee shop in the city centre . It is not child friendly. .
Diffusion-LM	The Twenty Two serves Indian food in the city centre . It is not family friendly .
FT	For low - priced food and a family - friendly atmosphere, visit Fitzbillies near Express by Holiday Inn
target length	22
FUDGE	The Golden Curry is an English food restaurant located near the Café Rouge in the Riverside area . The customer rating is average . Children are welcome .
Diffusion-LM	Strada is a fast food pub located near Yippee Noodle Bar and has a customer rating of 3 out of 5 .
FT	There is an Italian kid friendly restaurant in the riverside area near The Sorrento named Browns Cambridge in the riverside area .
target length	27
FUDGE	The Olive Grove is an expensive , children friendly , Fast food restaurant in the city centre . [missing 9 words]
Diffusion-LM	The Eagle is a family friendly coffee shop in the city centre near Burger King . It serves Italian food and has a low customer rating .
FT	A pub in the city centre near Yippee Noodle Bar is named Strada. It serves French food and has a customer rating of 3 out of 5
target length	32
FUDGE	The Golden Curry is a Japanese food restaurant with a high customer Rating , kid friendly and located along the riverside near Café Rouge . [missing 7 words]
Diffusion-LM	There is a family - friendly coffee shop in the city centre , it is called Zizzi . It is cheap and has a customer rating of 5 out of 5 .
FT	In the city centre is a kids friendly place called Green Man. It has Japanese food and is near All Bar One. It has a price range of £ 20 - 25
target length	37
FUDGE	There is a coffee shop called Fitzbillies which offers French food at cheap prices . It is not family - friendly and has a customer rating of 5 out of 5 . It is in riverside .
Diffusion-LM	The Rice Boat provides Indian food in the moderate price range . It is located in the city centre . It is near Express by Holiday Inn . Its customer rating is 3 out of 5 .
FT	For a family friendly coffee shop that serves Italian food, with a customer rating of 5 out of 5 and a cheap price range, try The Eagle. It is located in the riverside area .

Table 11: Qualitative output of the length control tasks, where all the generated texts tried to exactly match the target length. We mark the words exceeding the target length red.

target semantic content	name : Bibimbap House
FUDGE	Clare Hall , the Bibimbap House , serves high end Japanese food in the city centre .
Diffusion-LM	Bibimbap House in riverside near Clare Hall has a cheap price range .
FT	By Clare Hall is Bibimbap House which serves expensive noodles.
target semantic content	name : Travellers Rest Beefeater
FUDGE	Clowns near Clare Hall in riverside is a French coffee shop rated 5 out of 5
Diffusion-LM	Green Man is an Italian pub located in the city centre near Café UNK .
FT	Travellers Rest Beefeater is a reasonably priced restaurant that is family friendly.
target semantic content	Type : coffee shop
FUDGE	Wildwood is a coffee shop located near Ranch . It is expensive and highly UNK .
Diffusion-LM	The Punter is a high priced coffee shop located near Café Sicilia that serves Japanese food . It is not family - friendly and has a customer rating of 3 out of 5 .
FT	Located in the riverside area is the coffee shop Fitzbillies. It has Indian food in the price Range of less than £ 20 and a low customer Rating. It is not family Friendly.
target semantic content	customer rating : low
FUDGE	The Waterman is a fast food restaurant that is family - friendly near the city centre . [missing content]
Diffusion-LM	The Rice Boat restaurant has a low customer rating and is located in riverside . It serves Italian food , and is not family - friendly .
FT	The Eagle is low customer rating coffee shop with Italian food in riverside near Burger King. Its price range is less than £ 20 and is family - friendly.
target semantic content	near : The Sorrento
FUDGE	Browns Cambridge provides Indian food in the less than £ 20 price range . Its customer rating is low . [missing content]
Diffusion-LM	Near The Sorrento on the riverside is a pub named Taste of Cambridge that serves Japanese food .
FT	Browns Cambridge sells Italian food and is also a coffee shop. It has an average customer rating. It is located in the riverside area near Crowne Plaza Hotel and yes it is child friendly.
target semantic content	food : Italian
FUDGE	A non family - friendly Italian pub is Zizzi . It has an average customer rating .
Diffusion-LM	Loch Fyne is Italian Japanese restaurant that is kid friendly .
FT	situated near All Bar One is a child friendly Italian eatery called Green Man costing more than £ 30 is a restaurant near the riverside
target semantic content	price : high
FUDGE	The Vaults is a high priced Italian Pub with a customer rating of 3 out of 5 near Café Adriatic
Diffusion-LM	The Punter is a French restaurant with a high price range .
FT	A fast food coffee shop that is not kid friendly is called Cocum. It is expensive and gets average ratings.

Table 12: Qualitative output of the semantic content control task. We mark the compliant spans as green, and the spans that violates the control target as red.