Gradient-Based Constrained Sampling from Language Models

Sachin Kumar[♣] **Biswajit Paria**[♡] **Yulia Tsvetkov**[♠]
[♣]Language Technologies Institute, Carnegie Mellon University, Pittsburgh PA
[♡]Machine Learning Department, Carnegie Mellon University, Pittsburgh PA
[♠]Paul G. Allen School of Computer Science & Engineering, University of Washington, Seattle WA
{sachink,bparia}@cs.cmu.edu, yuliats@cs.washington.edu

Abstract

Large pretrained language models generate fluent text but are notoriously hard to controllably sample from. In this work, we study constrained sampling from such language models: generating text that satisfies user-defined constraints, while maintaining fluency and model's performance in a downstream task. We propose MUCOLA—a sampling procedure that combines the log-likelihood of the language model with arbitrary (differentiable) constraints in a single energy function, and then generates samples in a non-autoregressive manner. Specifically, it initializes the entire output sequence with noise and follows a Markov chain defined by Langevin Dynamics using the gradients of the energy function. We evaluate MUCOLA on text generation with soft and hard constraints as well as their combinations obtaining significant improvements over competitive baselines for toxicity avoidance, sentiment control, and keyword-guided generation.[1]

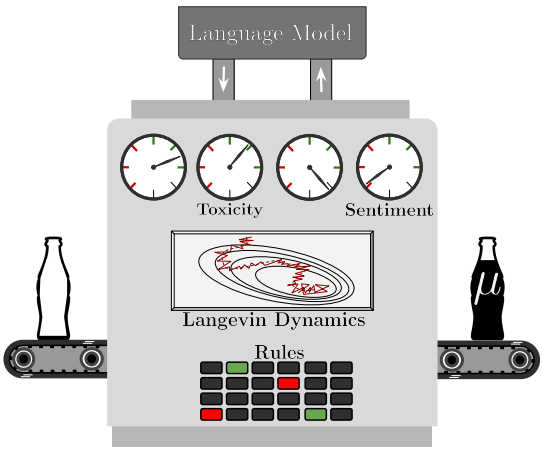

Figure 1: MUCOLA, our proposed method, stylized as μCOLA. Given a language model, a prompt/input $\mathbf{x}$, and desired constraints defined as thresholds on differentiable functions, we perform Langevin Dynamics updates to generate the entire output sequence $\mathbf{y}$ non-autogressively. We show experiments highlighting both hard and soft constraints (§4).

1 Introduction

Transformer-based language models (LMs) trained on web-scale corpora (Radford et al., 2019; Raffel et al., 2020; Brown et al., 2020) are generating impressively realistic texts. Despite having human-level fluency, they are far from reaching human-level communication abilities and are hard to control for content, context, and intent in communication. This results in unreliable models that lack basic knowledge, hallucinate facts, and discriminate users (Bender et al., 2021; Gehman et al., 2020; Pagnoni et al., 2021).

Controlled text generation—sampling text from LMs to satisfy constraints on the properties of generated text—aims to address these issues. Prior works incorporate constraints in existing decoding algorithms at token level by modifying output probabilities (Dathathri et al., 2020; Yang and Klein, 2021a; Krause et al., 2020a; Liu et al., 2021a; Lu et al., 2021b; Pascual et al., 2021; Liu et al., 2021b). While effective in certain settings, by generating autoregressively (i.e., left-to-right), these approaches fail to account for global context and hardly generalize beyond a single constraint. More importantly, by modifying output probabilities, they end up altering the underlying LM distribution compromising the fluency and task accuracy (Kumar et al., 2021).

We propose an algorithm to sample text *non-autoregressively* from a conditional or unconditional LM trained to perform any language generation task—translation, summarization, dialog, prompt completion—while controlling for multiple, potentially competing constraints, and without sacrificing the base model quality. Combining the LM likelihood with constraints into a single "energy" function, our algorithm follows a Markov

[1]The code is available at: https://github.com/Sachin19/mucoco/tree/sampling

Chain (Brooks et al., 2011) to iteratively transform an output sequence initialized randomly into a desired output with low energy.

Since common Monte Carlo Markov Chain (MCMC) sampling methods can be intractably slow (Sokal, 1997), we propose to define this Markov Chain using *gradients of the energy function* with respect to *token embeddings* of the output sequence. Additionally, we introduce stochasticity in the process in order to generate diverse samples, by modifying the gradients with additive noise, a process referred to as Langevin Dynamics (Grenander and Miller, 1994; Parisi, 1981; Welling and Teh, 2011; Gelfand and Mitter, 1991; Song et al., 2020). Finally, we operationalize the energy function by defining each constraint to be smaller than threshold, and writing it as a Lagrangian—with language model likelihood as the primary objective (Kumar et al., 2021). Besides allowing us to combine any number of constraints of varying scales without a need of tuning their weights, we show that low-energy solutions under this definition are true samples from the LM distribution. We call this algorithm MUCOLA for sampling with **mu**ltiple **co**nstraints from LMs using **La**ngevin Dynamics (§3; also see figure 1 in the Appendix).

We show the efficacy and generality of MUCOLA on a variety of tasks, LMs and constraints from prior work (§4), including soft constraints (§5) defined using auxiliary models (e.g., classifiers or smaller LMs), as well as hard rule-based constraints (§6) defined by presence or absence of certain keyphrases. We conduct experiments on (a) toxicity avoidance, sentiment control in GPT2-Large, and (b) keyword controlled generation with GPT2-Large and XL, and (c) entity control to improve fidelity in translation. Through both automatic metrics and human evaluation, we show versatility of this method through improved or matched performance with competitive baselines, in terms of quality and diversity. Finally, we present preliminary results on new tasks, showing promising directions for future research.

2 Background: Constrained Sampling from Language Models

Let $P(\mathbf{y}|\mathbf{x};\theta)$ model the conditional probability distribution of an output token sequence $\mathbf{y} = (y_1, \ldots, y_N)$, given an optional input token sequence $\mathbf{x} = (x_1, \ldots, x_M)$ where $x_m, y_n \in \mathcal{V}$.

We are interested in *constrained sampling* from P—finding output sequences $\mathbf{y}$ that have a high probability under P while minimizing a given set of constraint functions: $\{f_1, \ldots, f_C\}$. We assume that each $f_i : ([\mathbf{x}], \mathbf{y}) \rightarrow \mathbb{R}$ is defined such that a lower value of f_i implies that the output better satisfies the constraint. For example, to constrain the outputs to only non-toxic continuations for a given prompt $\mathbf{x}$, we define a classifier $p_{\text{TOXIC}}(\mathbf{y})$ which predicts the output toxicity probability, with lower probability implying lower toxicity. We assume all f_i are differentiable.

Enforcing these constraints in an autoregressive (i.e., left-to-right) decoding strategy like beam search or sampling is challenging, since the constraints are defined conceptually on the whole output sequence and are hard to evaluate accurately only on the generated prefix (Yang and Klein, 2021b; Liu et al., 2021a). With multiple constraints, their satisfaction/balancing becomes challenging. Recent work thus explored **non-autoregressive controlled generation** (Kumar et al., 2021), using constrained *optimization* over $\mathbf{y}$—finding a single output $\mathbf{y}$ which *maximizes* P given the constraints by performing gradient descent on the outputs $\mathbf{y}$. This involves (1) representing the constrained optimization problem as a single objective $\mathcal{E}(\mathbf{y})$ (often referred to as an *energy function*, discussed in §3.3), and (2) relaxing the discrete outputs $\mathbf{y}$ to continuous approximations such that gradient descent is feasible. In previous works, the latter is achieved by creating a soft-representation of $\mathbf{y}$, $\tilde{\mathbf{y}} = (\tilde{y}_1, \ldots, \tilde{y}_N)$ where each $\tilde{y}_n \in \mathbb{R}^{|\mathcal{V}|}$ is a simplex (or "logits" which are converted to a simplex using softmax) over the target vocabulary $\mathcal{V}$, representing the probability of the n-th token in the sequence. We refer to these methods as gradient-based decoding. Representing the decoding objective as $\min_{\tilde{\mathbf{y}}} \mathcal{E}(\tilde{\mathbf{y}})$ and initializing $\tilde{\mathbf{y}}$ with $\tilde{\mathbf{y}}^0$, it is updated as

$$\tilde{\mathbf{y}}^t = \tilde{\mathbf{y}}^{t-1} - \eta \nabla_{\tilde{\mathbf{y}}} \mathcal{E}(\tilde{\mathbf{y}}^{t-1}), \tag{1}$$

where $\eta > 0$ denotes the step size. In this process, the underlying LMs (and functions f_i) remain fixed and are used to provide gradients to the sequence $\tilde{\mathbf{y}}$. After performing multiple steps of this gradient descent, discrete text can be extracted from $\tilde{\mathbf{y}}$ using different heuristics (Kumar et al., 2021; Qin et al., 2020; Song et al., 2020). This formulation has been studied in various generation settings in prior work with different instantiations of $\tilde{\mathbf{y}}$ and $\mathcal{E}(\mathbf{y})$.

However, this setup is deterministic and does

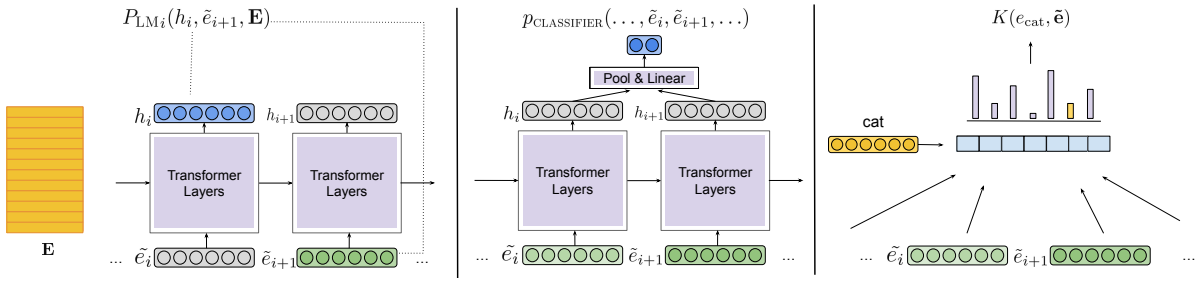

Figure 2: Different kinds of functions can be incorporated into MUCOLA defined on a shared embedding table $\mathbf{E}$. (Left) Language Modeling objective defines a per-token loss directly on the sequence of embeddings. For every token this loss provides gradients to update $\tilde{e}_i$ via backpropagation through the transformer layers and directly to $\tilde{e}_{i+1}$ through the negative loss likelihood loss as computed in §3.3. This is used as a primary objective for the underlying LM and can also be used for classification as discussed in §5.2 (Center) Classification objective defined on probability of the desired label. The classifier gets the token embeddings $\tilde{\mathbf{e}}$ directly as input and updates the embedding using gradients obtained via backpropagation from the transformer layers (Right) Lexical loss defined on the embeddings directly (without the use of additional models) to include desired keywords or phrases in the output sequence (§6). In practice any combination of these constraints can be used.

not facilitate sampling.[2] In addition, representing each token with a vector of size $|\mathcal{V}|$ can be computationally very expensive and difficult to fit into commonly used GPUs for long sequences (with more than ∼20-30 tokens; §7).

3 Constrained Sampling via Langevin Dynamics in Embedding Space

To enable efficient gradient-based sampling from LMs, we introduce MUCOLA which modifies the non-autoregressive framework in §2 to (a) generate multiple samples instead of optimizing for only one deterministic output, (b) optimize for much smaller intermediate token representations as opposed to their distribution on the entire vocabulary.

3.1 Exploring the token representation space

Instead of relaxing each target token y_n as a soft representation over the vocabulary $\tilde{y}_n \in \mathbb{R}^{|\mathcal{V}|}$, we represent it as $\tilde{e}_n \in \mathbf{E}$. Here $\mathbf{E}$ denotes the embedding table of the underlying language model containing $|\mathcal{V}|$ vectors of size $d \ll |\mathcal{V}|$. We denote this sequence of embeddings as $\tilde{\mathbf{e}} = \{\tilde{e}_1, \ldots, \tilde{e}_N\}$. At an update step t, instead of feeding each $\tilde{\mathbf{y}}$ to the model(s) (which are then transformed to an embedding to be fed to the first layer), we directly feed $\tilde{\mathbf{e}}$ to the first layer to compute the energy function, now defined as a function of embeddings instead of tokens. In case of deterministic minimization (similar to (1)), these vectors are updated as

$$\tilde{\mathbf{e}}^t = \text{Proj}_{\mathbf{E}}(\tilde{\mathbf{e}}^{t-1} - \eta \nabla_{\tilde{\mathbf{e}}} \mathcal{E}(\tilde{\mathbf{e}}^{t-1})), \qquad (2)$$

where $\text{Proj}_{\mathbf{E}}(\hat{e}) = \arg\min_{e \in \mathbf{E}} \|e - \hat{e}\|_2$ denotes a projection operation on the embedding table $\mathbf{E}$. In other words, after every gradient step, we project each updated vector back to a quantized space, that is the embedding table using Euclidean distance as the metric. This projection is done to prevent adversarial solutions.[3] After the optimization is complete, discrete text can be easily obtained by projection, that is the token indices corresponding to each $\tilde{e}_n$ in the embedding table $\mathbf{E}$. This formulation yields the following benefits: (a) For a sequence of length L, at any optimization step t, it only maintains (and computes gradients with respect to) $L \times d$ parameters, as opposed to $L \times |\mathcal{V}|$. This enables us to store much longer sequences in a GPU as compared to the storing $\tilde{\mathbf{y}}$. (b) this formulation provides a natural way to define hard rule-based constraints based on keywords or phrases (discussed in more detail in §6), and, finally (c) it yields a natural way to generating samples.

3.2 Gradient based Sampling via Langevin Dynamics

The minimization in (2) can be very easily extended to a sampling procedure by modifying the gradient descent in (2) to Langevin Dynamics (Gelfand and Mitter, 1991; Welling and Teh, 2011),

$$\tilde{\mathbf{e}}^t = \text{Proj}_{\mathbf{E}}(\tilde{\mathbf{e}}^{t-1} - \eta \nabla_{\tilde{\mathbf{e}}} \mathcal{E}(\tilde{\mathbf{e}}^{t-1}) + \sqrt{2\eta\beta} z^t)$$

[2] While initialization can be used to add randomness to this algorithm, we find that it has little to no effect on diversity.

[3] Several prior works (Belinkov and Glass, 2019) have shown that neural-network based models are not robust to change in input space. We observed this phenomenon in our preliminary experiments where, without any projection, most low energy solutions were found to be garbled text.

Langevin Dynamics provides a MCMC method to sample from a distribution using only the gradient of its logarithm. That is, if we define a distribution as $Q(\mathbf{y}) \propto \exp(-\mathcal{E}(\mathbf{y}))$, its logarithm leads to the update specified above.[4] This method is often used for non-convex optimization for training neural networks (Welling and Teh, 2011) due to its ability to escape local minima due to added noise and converge towards the global minima. In this work, we adapt it for inference (Song and Ermon, 2019).

Intuitively, by adding noise at every gradient step, this procedure intends to find outputs $\mathbf{y}$ that do not exactly minimize $\mathcal{E}$ but remain in the vicinity of the minima. In other words, it finds outputs which admit high probability under the distribution $Q(\mathbf{y})$. This process begins with an exploration phase which is controlled by β. With a high value of β, the noise term is large leading to big updates. By gradual annealing such that $\beta \to 0$, as $t \to \infty$, this process converges to a sample from $Q(\mathbf{y})$.[5]

3.3 Representing the energy function

A straightforward way to represent $\mathcal{E}$ is with a linear combination as $\sum_{i=1}^{C} \lambda_i f_i - \lambda_{C+1} \log P$, with pre-defined weights $\lambda_1, \ldots, \lambda_{C+1}$ (Hoang et al., 2017; Qin et al., 2020, 2022). With this formulation (a) linear weights, λ_i's, can be hard to define and tune for different f_i, and especially difficult when f_i's lie on different scales, and more importantly, (b) defining the energy function in this manner modifies the original goal, which is to sample from the language model P (with constraints), not from a modified distribution $Q \propto \exp(-\mathcal{E}(\mathbf{y}))$. To alleviate these issues, we define the inference objective, following Kumar et al. (2021), as

$$\mathbf{y} \sim P(\mathbf{y}|\mathbf{x};\theta), \text{ subject to } f_i([\mathbf{x}], \mathbf{y}) \leq \epsilon_i \forall i$$

where each threshold ϵ_i is a hyperparameter. As we discuss in more detail in §4, these thresholds can be flexibly defined for most kinds of constraints. For example, instead of merely trying to reduce $p_{\text{TOXIC}}(\mathbf{y})$, we can set it as $p_{\text{TOXIC}}(\mathbf{y}) < 0.1$. Given this formulation, we define the energy function as a Lagrangian $\mathcal{E}(\mathbf{y}) = -\log P(\mathbf{y}) - \sum_{i=1}^{u} \lambda_i(\epsilon_i - f_i(\mathbf{y}))$. Here, $\lambda_i \geq 0$ are Lagrangian multipliers and dynamically updated at each step. We follow the gradient of $\mathcal{E}$ downwards for the $\tilde{\mathbf{e}}$ (as described in (2)) and upwards for the multipliers (gradient ascent without any noise) while making sure that the multipliers remain positive: $\lambda_i^t = \max(0, \lambda_i^{t-1} + \alpha \nabla_{\lambda_i} \mathcal{E}(\mathbf{y}))$ ($\alpha > 0$ is the step size for ascent). Intuitively, if a constraint is not satisfied, the term $(\epsilon_i - f_i(\cdot))$ would be negative and λ_i would keep increasing making $\mathcal{E}$ high. On the other hand, if all the constraints are satisfied these values gradually decrease to 0 making $\mathcal{E}(\mathbf{y}) = -\log P(\mathbf{y})$ making the final output a sample from the desired distribution P. We implement a damped version of this process to improve stability, the details of which can be found in Kumar et al. (2021). The final decoding algorithm we used in our experiments is described in algorithm 1 in the Appendix.

Energy as a function of embeddings Performing gradient updates with respect to $\tilde{\mathbf{e}}$ requires that all objectives be defined as functions of $\tilde{\mathbf{e}}$, not $\mathbf{y}$. Also, $f_1(\mathbf{y}), \ldots, f_C(\mathbf{y})$ must share the same input embedding table (as that of P). We discuss in §4 how this can achieved for different kinds of constraint functions f_i. First, we describe how to compute the primary objective $-\log P(\mathbf{y}|\mathbf{x};\theta)$ and its gradients with respect to $\tilde{\mathbf{e}}$. In typical LMs, this objective is factorized as $\log P(\mathbf{y}|\mathbf{x}) = \sum_{n=0}^{L-1} \log P(y_{n+1}|y_{1:n}, \mathbf{x})$. For each decoding step $n+1$: the model takes as input y_n, which is converted to e_n via an embedding table lookup. Passed through the network layers, it is converted to a hidden vector h_n. Since the input and output embedding tables in most modern LMs are shared (Radford et al., 2019; Raffel et al., 2020; Lewis et al., 2020; Brown et al., 2020),[6] the softmax probability is computed as, $P(y_{n+1}|y_{1:n}, \mathbf{x}) = \frac{\exp(h_n^T e_{n+1} + b_{n+1})}{\sum_{j=1}^{|\mathcal{V}|} \exp(h_n^T e_j + b_j)}$ where b_n are optional bias terms. By replacing e_{n+1} with $\tilde{e}_{n+1}$, we convert the above probability to $P(\tilde{e}_{n+1}|\tilde{e}_{1:n}, \mathbf{x})$. For each position $n+1$, $\tilde{e}_{n+1}$

[4] The normalization term in $Q(\mathbf{y})$ vanishes as its gradient with respect to $\mathbf{y}$ is 0.

[5] More details of the implementation of annealing schedule can be found in §4. A similar noise can also be applied directly to the soft-token representations in (1) as explored in Qin et al. (2022). However, as we discuss in §7, our formulation with its smaller parameter size allows generating longer sequences. In addition, considering logits as soft-representations (followed by softmax) has shown to result in slow mixing, that is, it takes much longer to converge as empirically shown in Hoang et al. (2017) and also observed in Qin et al. (2022). On the other hand, considering the simplex itself (Kumar et al., 2021; Hoang et al., 2017) as soft-representations is not compatible with Gaussian noise and can lead to undesirable behavior (Patterson and Teh, 2013).

[6] Even if the embedding tables are not shared, this loss may be computed and optimized using vectors from the output embedding table as parameters without any significant loss in performance.

receives gradients, (a) directly from $-\log P$ function and (b) through h_{n+1} via back-propagation through the network layers (See figure 2 (left)).

4 Experimental Setup

We evaluate MuCoLa on four constrained generation tasks. These tasks are selected based on defining different kinds of constraints for which prior work designed specialized training or decoding mechanisms which cannot be generalized beyond those tasks or language models. The main contribution of MuCoLa is generating diverse samples which conform to the language model P as well as can satisfy user defined arbitrary combination of constraints for which fine-tuning is generally infeasible and tuning weights of each constraint is cumbersome. For a pre-defined sentence length L, we initialize the token representation for each step $\tilde{e}_1, \ldots, \tilde{e}_L$ using token embeddings randomly sampled from the target vocabulary $\mathcal{V}$.[7] For all our experiments, we run the Langevin Dynamics simulation for a maximum of 250 iterations unless specified otherwise. We describe additional implementation details including noise schedule, stopping criterion and multiplier update schedule in Appendix C.

5 Text Generation with Soft Constraints

First, we evaluate MuCoLa with real valued constraint functions defined via auxiliary models such as classifiers or LMs. Given an LM GPT2-Large (Radford et al., 2019), and a prompt $\mathbf{x}$, we generate continuations $\mathbf{y}$. We conduct experiments with: toxicity avoidance and sentiment control. Each of the tasks define a binary constraint. Let the desired label be denoted by LABEL_1, and other one with LABEL_0 (LABEL_1 is non-toxic in toxicity avoidance and positive in sentiment control). For both setups, we assume availability of corpora to train the constraint functions.[8]

Baselines In addition to decoding without any constraints (which we simply call GPT2), we consider the following baselines which decode from left-to-right:

- **Domain Adaptive Pretraining (DAPT)** (Gururangan et al., 2020) proposes to finetune the LM P on a corpus of desired constraint and sample directly from finetuned version.
- **FUDGE** (Yang and Klein, 2021a) uses a "future-aware" constraint classifier to modify output token probabilities at every decoding step to steer the generation to promote constraint satisfaction. This classifier is trained to predict the ground truth label for every prefix of the training corpus.
- **GeDi** (Krause et al., 2020a) uses a class-conditioned LM to modify output token probabilities at each step via Bayes' rule.
- **DExperts** (Liu et al., 2021b) proposes to replace the class-conditioned LM with two auxiliary language models (one expert and one anti-expert) to modify the output logits at every step. These LMs are trained using same setup as the baseline **DAPT** but instead of sampling from them directly, it uses them to steer the base LMs outputs. For each of the baselines, we use recommended hyperparameters to generate samples.

Constraint functions Each of these baselines can be adopted as constraint functions for MuCoLa as follows:

- **Discriminative Classifiers** We train a binary classifier $p_{\text{LABEL}}(\mathbf{y})$, which predicts the probability of the desired attribute given the output sequence $\mathbf{y}$ by finetuning roberta-base with GPT2-Large embeddings (more details in Appendix D). To decode with MuCoLa, we formulate this constraint as $p_{\text{LABEL}_1} > \epsilon$ (We define specific ϵ values in §5.1 and §5.2 respectively). To improve its gradient profile, we use the constraint in log space. We call this setup **MuCoLa-DISC**.
- **Generative Classifiers** Prior work has shown that discriminative classifiers can be fragile to domain shift or adversarial examples (Yogatama et al., 2017; Krause et al., 2020b). Hence, we also consider a second class of *generative* classifiers trained as class conditional LMs that model $p(\cdot|\text{LABEL})$. Intuitively, they are required to explain every word in the input, potentially amplifying the class signal and improving robustness (Min et al., 2021). We define them in three ways by finetuning GPT2-Large: (1) following GeDi (**MuCoLa-GeDi**), (2) following DEx-

[7] We also tried other initialization strategies like initializing with zeros, or outputs of nucleus sampling or greedy decoding but did not find it to have any significant effect on the final output

[8] This setup can be easily extended to n-class setups by defining $n-1$ constraints as $p_0 > p_1, \ldots, p_0 > p_{n-1}$

perts, (we train two separate LMs; **MuCoLa-DExperts**). And finally, (3) motivated by recent work on prompt-based classification, we define a class-conditional LM without finetuning the model as $P(\mathbf{x}, \mathbf{y}|\text{verbalize}(\text{LABEL}))$ where verbalize$(\cdot)$ is function that converts the label to a natural language string (**MuCoLa-prompt**). Note that for all three setups, the embedding table is frozen (more details in Appendix D). We decode via MuCoLa with the constraint $p(\mathbf{x}, \mathbf{y}|\text{LABEL}_1)) > p(\mathbf{x}, \mathbf{y}|\text{LABEL}_0)$ (again, realized in log-space)[9].

Evaluation In both experiments, we evaluate the generated samples along three dimension, (1) **Constraint Satisfaction** measured using external evaluators, (2) **Fluency**, measured by mean perplexity of the continuations measured using GPT2-XL. Since the objective is to generate samples from the LM, we rank different methods not by their absolute perplexity, but its difference from the perplexity of unconstrained text. Additionally, we also report a grammaticality score: the fraction of outputs predicted by a classifier trained on CoLA (Warstadt et al., 2019) as fluent. (3) **Diversity**, measured by computing the mean number of distinct n-grams in each set of samples, normalized by the length of text (Li et al., 2016). We report this for $n = 1, 2, 3$ following prior work. Since all the automatic metrics are model based and can be biased, we also perform human evaluation in an A/B testing setup with the best performing baseline (DExperts in our case). For each sample, we ask 3 annotators to compare and rank candidates from our approach and the baseline on constraint satisfaction, topicality and fluency.

5.1 Toxicity Avoidance

Prior work have shown that large pre-trained LMs are at risk of producing toxic content even when given innocuous prompts (Sheng et al., 2019; Gehman et al., 2020). In this experiment, given a neutral prompt, we generate non-toxic continuations using MuCoLa. We only consider the setup MuCoLa-DISC here, with a classifier p_{TOXIC}, trained on a dataset of human-annotated comments labelled as toxic or non-toxic (Appendix D). We decode with the constraint $p_{\text{TOXIC}} < 0.01$.

We follow the evaluation setup defined in Liu et al. (2021b) and use test set of 10K nontoxic prompts (Gehman et al., 2020) where without any constraints, the user might receive harmful output from the LM. For each prompt, we generate 25 samples for length 20 tokens each. We measure constraint satisfaction using the toxicity score from Perspective API. Following prior work (Gehman et al., 2020; Liu et al., 2021b), we report the maximum toxicity score over 25 samples per prompt averaged over the number of prompts, and the empirical probability of generating a continuation with toxicity > 0.5 at least once over the 25 generations.

As shown in Table 1, MuCoLa outperforms or matches all baselines on toxicity, including a strong baseline DExperts which is specifically designed for binary constraints. In addition, our method is closest in perplexity to unconstrained generation, while maintaining grammaticality as well as diversity of baseline methods[10]. We attribute this improvement to the fact that after the constraints are satisfied, the energy function in MuCoLa reduces to $-\log P(\mathbf{y})$, the original function we intend to sample from, whereas in the baselines, the underlying probability distribution (or the energy function) is modified to achieve control. Furthermore, human evaluation (Appendix F) reveals that generations by MuCoLa match DExperts on toxicity and fluency while being more topical.

5.2 Sentiment Controlled Generation

Given a prompt $\mathbf{x}$, the goal of this task is to generate continuations $\mathbf{y}$ using an LM with a positive sentiment/polarity. To understand the effect of sources of training data, we train two versions of each constraint function described above on two datasets: SST-2 corpus (Socher et al., 2013) containing $\sim$4K examples in Movie reviews for each class; and Yelp polarity corpus containing $\sim$280K examples for each class containing a mixed domain of reviews. We also consider an additional setup where we use two constraints using both versions of **MuCoLa-DISC**, which we call **MuCoLa-TWO-DISC**. We use a dataset of 15 prompts from Dathathri et al. (2020) and generate 20 samples per prompt of length 12, 20, and 50.

To evaluate constraint satisfaction, we measure positive sentiment accuracy of the output using

[9] Note that all constraints we descibe can be easily extended to n-class set (with say 0 as the desired label) by defining $n-1$ constraints as $p_0 > p_1, \ldots, p_0 > p_{n-1}$

[10] While FUDGE obtains the lowest absolute perplexity, prior work (Holtzman et al., 2020) has shown that very low perplexity is not an indicator of higher quality but of repetitions and usage of only high frequency tokens.

Approach	Toxicity		Fluency		Diversity		
	Avg. Max. Toxicity	Toxicity Prob.	Perplexity	CoLa Accuracy	Dist-1	Dist-2	Dist-3
GPT-2	0.527	0.520	25.45	88.3	0.58	0.85	0.85
DAPT	0.428	0.360	31.21	**91.2**	0.57	0.84	0.84
FUDGE	0.437	0.371	12.97	88.5	0.47	0.78	0.82
GEDI	0.363	0.217	60.03	85.5	0.62	0.84	0.83
DEXPERTS	**0.302**	0.118	38.20	89.8	0.56	0.82	0.83
MUCOLA	**0.308**	**0.088**	**29.92**	88.2	0.55	0.82	0.83

Table 1: Results for toxicity avoidance (§5.1). We evaluate on three axes: (1) Toxicity–Avg. Max. Toxicity and Toxicity Prob.: lower the better. (2) Fluency–GPT2-XL Perplexity: the closer the value to unconstrained outputs (GPT2: 38.6), the better; CoLa accuracy: higher the better, and (3) Diversity (Dist-1,2,3): higher the better. The best values in each column are highlighted in **bold**. While our method improves or performs on par with baselines on toxicity metrics, we obtain substantial improvements on perplexity.

three external classifiers to account for domain differences in their training data, (a) **C1**: distilbert (Sanh et al., 2019) finetuned on SST-2 data, used in (Liu et al., 2021b), (b) **C2**: bert-base (Devlin et al., 2019) finetuned on Yelp Polarity corpus used in Mireshghallah et al. (2022), and (c) **C3**: SieBERT (Heitmann et al., 2020) finetuned on 15 different polarity datasets. We report a subset of results of this experiment in table 2 for outputs of length 20 (and the rest in Appendix F). We observe a significant variance in sentiment control accuracies (C1, C2 and C3) where constraints trained on SST-2 perform worse on the evaluator trained on Yelp (C2) and vice versa for all methods. The third evaluator (C3) trained on a much larger training set can be considered more reliable. Overall, we find that MUCOLA in all settings obtains perplexity values closer to unconstrained outputs (GPT2) whereas most baselines achieve control at the cost of perplexity. Surprisingly, constraints trained on Yelp perform poorly compared to those trained on SST2 despite the former being a larger dataset.

For outputs of lengths 12 and 20, MUCOLA-TWO-DISC finds a good balance of control and fluency and outperforms all other baselines on sentiment accuracy while maintaining good perplexity (except GEDI which performs poorly on perplexity as well as CoLa accuracy). This improvement however comes with a slight decline in diversity metrics which we argue is a fair price to pay for constraint satisfaction compared to fluency. Similar to §5.1, a small scale study on human evaluation (Appendix F) reveals MUCOLA to be more topical than the best baseline DExperts. Finally, using a prompt-based constraint also performs strongly despite not trained at all. In future work, we will look into training a prompt-based classifier to improve this performance. For outputs of length 50, we observe a slight drop in MUCOLA's performance. On closer inspection (table 14), we find a trend of degenerate repetitions at the end of many sequences. Prior work (Holtzman et al., 2020) has shown that large LMs often assign unusually high probabilities to repeating sequences especially with increasing lengths and since our method is designed to sample high probability outputs, such behavior is expected. In future work, we will explore constraints designed to discourage this behavior (Welleck et al., 2020; Meister et al., 2022).

6 Decoding with Hard Constraints

In the previous two tasks, we explored how MUCOLA can be applied on soft constraints, defined via real valued functions like probabilities of classifiers or language models. Now, we consider a ruled-based constraint that a specific word or phrase *must* appear in the generated text. Existing autoregressive solutions to this task have explored various strategies either based on explicitly modifying probabilities to up-weight desired words (Pascual et al., 2021), or search-based strategies based on beam-search (Lu et al., 2021b). In this work, we define a differentiable distance function $d(w, \tilde{\mathbf{e}})$ which measures overlap between desired word (w) and the output token embeddings $\tilde{\mathbf{e}}$ (we use the notation w to refer to as the word itself and its index in the vocabulary interchangeably). We then propose a simple criterion to define a threshold ϵ that guarantees that if $d(w, \tilde{\mathbf{e}}) < \epsilon$, then w's embedding appears in $\tilde{\mathbf{e}}$ (and by extension w appears in $\mathbf{y}$). Inspired from Liu et al. (2022); Qin et al. (2022), this distance is computed in three steps (1)

define a "probability distribution" for each output token, $\pi_n = \text{softmax}(-\|\tilde{e}_n - e_1\|_2^2, \ldots, -\|\tilde{e}_n - e_{|\mathcal{V}|}\|_2^2)$ where $\{e_1, \ldots, e_{|\mathcal{V}|}\} \in \mathbf{E}$, (2) Define $q = \text{gumbel} - \text{softmax}(g_1/\tau, \ldots, g_N/\tau)$, where $g_n = \log \pi_{n,w}$, gumbel softmax provides a way to differentiably sample (Jang et al., 2017) and τ is a hyperparameter, and finally, (3) $d(w, \tilde{\mathbf{e}}) = \sum_{n=1}^{N} -q_n g_n$. Intuitively, this function minimizes the Euclidean distance between one of the output embeddings (chosen with stochasticity) and w's embedding, e_w. This function can be easily extended for phrases, $w = (w_1, \ldots, w_l)$ by defining $g_n = \frac{1}{l} \sum_{u=1}^{l} \log \pi_{w_u, n+u}$.

Based on this definition, for each desired keyword, we define a threshold ϵ_w as $-\log \pi_{w,w}$.We provide an intuitive explanation of the distance function, and a semi-formal and empirical proof of hard satisfaction of this threshold in Appendix E.

Tasks We formally evaluate this setup on two tasks: (1) open-ended keyword constrained generation (with two datasets: COMMONGEN and ROC)), and (2) terminology constrained machine translation. We additionally show preliminary findings on a third task, entity guided summarization. We elaborate on COMMONGEN here and the rest of the results, following a similar trend can be found in Appendix E. In COMMONGEN (Lin et al., 2020) given no prompt , the task is generate an output of maximum length 40 which contains a given set of four or five words. We use GPT2-XL as the underlying LM in this setup with COLD (Qin et al., 2022) as our main baseline.

Evaluation Following prior work, we measure the performance on two axes, (1) **Coverage**, measured by (a) count average number of keywords appearing in the output; and (b) percent, measuring the fraction of outputs which contain all the desired keywords. (2) **Fluency**, as measured by GPT2-XL perplexity and human evaluation, where on a sample of 200 outputs, we ask 3 annotators to rate each output on a 3-point likert scale. As reported in table 3, we outperform the best baseline on coverage. We outperform all baselines in terms of perplexity by a large margin, again owing to the fact that our method samples from the language model and does not modify the distribution itself as opposed to the baselines. Human evaluation reveals that our approach slightly underperforms the best baseline.

7 Discussion and Analysis

Speed and Memory Generating a sequence of length L using MUCOLA requires maintaining $L \times d$ parameters. In contrast, performing Langevin Dynamics in the vocabulary space requires $L \times |\mathcal{V}|$ parameters ($|\mathcal{V}| >> d$). In this analysis, we empirically verify the benefits of our setup. With GPT2-Large as underlying LM, we sample sequences of varying lengths with various constraints, on different commercially available GPUs using both our approach and an ablation with vocabulary sized representations (logits+softmax; more details in Appendix H). As summarized in table 10, we find that much longer sequences can be generated with embeddings across the board (maximum of length 500 even with constraints) while with vocabulary sized parameters, the approach runs of out of memory even without any constraint beyond a length of 20 even on the largest GPU.

Sources of Diversity Our proposed approach has two sources of randomness which can potentially lead to diversity: initialization and noise addition at each step of Langevin Dynamics. To understand their effects, we vary these sources and compute the diversity metrics. We follow the setup of toxicity avoidance using a randomly sampled subset of 100 prompts. The results are shown in table 8. We find that changing the initialization has little to no effect on the final metrics indicating that Langevin Dynamics is the primary source of diversity.

Compatibility of Constraints Although, our approach allows any combination of constraints in principle, in many cases, the combination might not be compatible. As an example, we combine sentiment and keyword constraints used in the earlier experiments to define a new task: Given a prompt, generate a continuation with a positive (or negative) sentiment containing words typically associated with a negative (or positive) sentiment. Using our best performing constraint (MUCOLA-TWO-DISC) from §5.2, and a single keyword constraint, we find that MUCOLA fails almost $\sim$90% of the times since two constraints are incompatible for most scenarios. For when it does succeed, we present selected examples in table 19.

Varying threshold ϵ In our experiments, each function f_i is constrained to be bounded by a thresholds ϵ_i, which are tunable hyperparameters. The threshold provides an interpretable way to control

Approach	Setting	% Positive Sentiment			Fluency		Diversity		
		C1	C2	C3	Perplexity	CoLa	Dist-1	Dist-2	Dist-3
GPT-2	-	46.7	47.7	61.3	38.6	78.7	0.64	0.90	0.88
DAPT	SST-2	73.6	70.0	78.3	76.9	70.7	0.64	0.89	0.86
FUDGE	SST-2	67.6	63.0	79.3	10.3	**94.0**	0.51	0.80	0.84
GEDI	SST-2	99.0	96.3	99.7	268.7	54.0	0.69	0.87	0.84
DEXPERTS	SST-2	91.2	83.4	95.4	55.37	81.6	0.61	0.89	0.87
MUCOLA-DISC	SST-2	84.6	77.5	88.0	27.9	80.8	0.50	0.81	0.82
MUCOLA-DISC	Yelp	83.0	83.6	83.0	**32.2**	76.0	0.50	0.75	0.80
MUCOLA-TWO-DISC	Yelp, SST-2	**93.7**	**91.0**	**96.0**	28.9	76.7	0.53	0.77	0.74
MUCOLA-PROMPT	-	87.3	91.0	93.0	53.0	77.2	0.54	0.82	0.80

Table 2: Results for Sentiment Controlled Generation for outputs of length 20. We evaluate on three axes: (1) % Positive Sentiment: higher the better. We use three external classifiers for this evaluation, C1 trained on SST2 data, C2 trained on Yelp data, and C3 trained on 15 polarity datasets; (2) Fluency–GPT2-XL perplexity, closer the value to unconstrained outputs (GPT2: 38.6), the better; CoLa accuracy: higher the better, and (3) Diversity (Dist-1,2,3): higher the better. The best values in each column are highlighted in **bold**.

	Coverage		Fluency	
	Count	Percent	Perplexity	Human
TSMH	2.72	71.27	1545.15	1.72
Neurologic	3.30	91.00	28.61	**2.53**
COLD	4.24	94.5	54.98	2.07
MUCOLA	**4.49**	**99.7**	**23.50**	2.29

Table 3: Results of keyword constraint on COMMONGEN. We report (a) coverage as avg. count of desired keywords in the output and the fraction of the outputs containing all keywords (percent); and (b) GPT2-XL perplexity and avg. fluency score rated by humans.

the intensity of the desired attributes. To illustrate this capability, we again follow the setup of toxicity avoidance with 100 prompts and apply the constraint $p_{\text{TOXICITY}} < \epsilon$ with $\epsilon \in \{0.5, 0.3, 0.1, 0.01\}$. As shown in table 8, making ϵ smaller improves toxicity control. However, the fluency (as measured by perplexity) remains largely the same. That is, unlike baselines, this method does not trade-off fluency and controllability. However, there is a trade off between diversity and controllability as we observe in sentiment control experiments (§5.2) where making a constraint stricter leads to a decline in diversity.

8 Conclusion

We present MUCOLA, a sampling algorithm from language models that flexibly combines pretrained LMs with any differentiable constraints. Our primary contributions are a (1) gradient based MCMC sampling method (Langevin Dynamics) performed on (2) intermediate representation of tokens (embeddings). With experiments on both soft and hard constraints with different pretrained LMs, we show that this approach generates diverse outputs which better conform both to desired constraints as well as the underlying LM distribution. Despite the observed improvements, we believe we have barely scratched the surface. In future work, we will explore ways to improve the convergence properties of this algorithm using more sophisticated MCMC algorithms (Girolami and Calderhead, 2011) and develop constraints to improve performance on longer sequences. Furthermore, since we perform updates on embeddings rather than vocabulary distributions, future work may also study ways to expand vocabularies at decoding time.

Acknowledgements

The authors would like to thank Xiaochuang Han, Artidoro Pagnoni, Melanie Sclar, Lucille Njoo and Anjalie Field for helpful feedback and discussions, and the anonymous reviewers for much appreciated feedback. S.K. is supported by a Google PhD Fellowship. This material is based upon work supported by the National Science Foundation under CAREER Grant No. IIS2142739, as well as Grants No. IIS2125201 and IIS2203097. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

Limitations

Despite speed improvements on gradient-based decoding compared to using vocabulary-sized representation, this approach still requires iteratively updating $L \times d$ parameters (with each update in-

volve a forward and a backward pass) and is considerably slower than autoregressive decoding methods (anywhere between 15-20 times longer). A straightforward way to improve decoding speed is using larger batches and smaller floating point operations which we leave for future work. Further improvements may also be achieved by adapting more sophisticated gradient based methods for faster convergence (Girolami and Calderhead, 2011) or techniques from diffusion models in image generation (Luhman and Luhman, 2021). Like other non-autoregressive decoding approaches, this method also requires pre-defining a fixed output length which can be a hindrance. This is an active area of research with many solutions proposed in the literature including predicting the sequence length (Wang et al., 2021), generating multiple outputs with varying lengths and reranking (Guo et al., 2019), continuing generating autoregressively to finish a sentence after a fixed length output is generated (Qin et al., 2022) of all which have shown promising results. Furthermore, since this algorithm aims to find high probability sequences under the LM, and most open-ended LMs suffer from degeneracy (repeating sequences get high probability) (Holtzman et al., 2020), using it can sometimes lead to such generations especially for long sequences (as we observe in §5.2). Incorporating repetition reducing loss functions (Welleck et al., 2020; Meister et al., 2022) as constraints can help alleviate this issue. We leave this exploration for future work.

Ethical Considerations

Most large LMs trained on web content have been shown to generate harmful language, generating toxic and non-factual content (Gehman et al., 2020; Pagnoni et al., 2021), and can potentially be used maliciously (Wallace et al., 2019, 2020; Zellers et al., 2019). The ultimate goal of controlled text generation approaches, including ours, is to enable finer-grained control over generated texts that could potentially alleviate these issues (Gehman et al., 2020; Bender et al., 2021; Liu et al., 2021a). They also find useful applications in anonymizing protected attributes in written text (Reddy and Knight, 2016), as writing assistants to avoid users' implicit biases (Ma et al., 2020; Field and Tsvetkov, 2020). However, none of the approaches are perfect.

On the other hand, controlled text generation research can also be maliciously used to generate misinformation, make output text more biased and toxic as well as target individuals to influence public opinion. To combat these issues, future research should focus on developing better defense methods against misusing these models maliciously, in a way that could cause societal harms (Zellers et al., 2019).

References

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proc. FAccT*.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Raphael Rubino, Lucia Specia, and Marco Turchi. 2017. Findings of the 2017 conference on machine translation (WMT17). In *Proceedings of the Second Conference on Machine Translation*, pages 169–214, Copenhagen, Denmark. Association for Computational Linguistics.

Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng. 2011. *Handbook of Markov Chain Monte Carlo*. CRC press.

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. 2020. Language models are few-shot learners. In *Proc. NeurIPS*.

Fredrik Carlsson, Joey Öhman, Fangyu Liu, Severine Verlinden, Joakim Nivre, and Magnus Sahlgren. 2022. Fine-grained controllable text generation using non-residual prompting. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6837–6857, Dublin, Ireland. Association for Computational Linguistics.

Alvin Chan, Yew-Soon Ong, Bill Pung, Aston Zhang, and Jie Fu. 2021. Cocon: A self-supervised approach for controlled text generation. In *Proc. ICLR*.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and

Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *Proc. ICLR*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Georgiana Dinu, Prashant Mathur, Marcello Federico, and Yaser Al-Onaizan. 2019. Training neural machine translation to apply terminology constraints. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3063–3068, Florence, Italy. Association for Computational Linguistics.

Anjalie Field and Yulia Tsvetkov. 2020. Unsupervised discovery of implicit gender bias. In *Proc. EMNLP*.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A. Smith. 2020. RealToxicityPrompts: Evaluating neural toxic degeneration in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3356–3369, Online. Association for Computational Linguistics.

Saul B. Gelfand and Sanjoy K. Mitter. 1991. Recursive stochastic algorithms for global optimization in $\mathbb{R}^d$. *SIAM Journal on Control and Optimization*.

Mark Girolami and Ben Calderhead. 2011. Riemann manifold langevin and hamiltonian monte carlo methods. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*.

Ulf Grenander and Michael I. Miller. 1994. Representations of knowledge in complex systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 549–603.

Junliang Guo, Xu Tan, Di He, Tao Qin, Linli Xu, and Tie-Yan Liu. 2019. Non-autoregressive neural machine translation with enhanced decoder input. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):3723–3730.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Mark Heitmann, Christian Siebert, Jochen Hartmann, and Christina Schamp. 2020. More than a feeling: Benchmarks for sentiment analysis accuracy. *Available at SSRN 3489963*.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. 2022. Video diffusion models. *arXiv:2204.03458*.

Cong Duy Vu Hoang, Gholamreza Haffari, and Trevor Cohn. 2017. Towards decoding as continuous optimisation in neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 146–156, Copenhagen, Denmark. Association for Computational Linguistics.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparametrization with gumbel-softmax. In *Proceedings International Conference on Learning Representations (ICLR)*.

Vivek Jayaram and John Thickstun. 2021. Parallel and flexible sampling from autoregressive models via langevin dynamics. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 4807–4818. PMLR.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. 2018. Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, Melbourne, Australia. Association for Computational Linguistics.

Nitish Shirish Keskar, Bryan McCann, Lav Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL - A Conditional Transformer Language Model for Controllable Generation. *arXiv preprint arXiv:1909.05858*.

Muhammad Khalifa, Hady Elsahar, and Marc Dymetman. 2021. A distributional approach to controlled text generation. In *International Conference on Learning Representations*.

Tomasz Korbak, Hady Elsahar, German Kruszewski, and Marc Dymetman. 2022. Controlling conditional language models without catastrophic forgetting. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 11499–11528. PMLR.

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

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. 2020b. GeDi: Generative discriminator guided sequence generation.

Sachin Kumar, Eric Malmi, Aliaksei Severyn, and Yulia Tsvetkov. 2021. Controlled text generation as continuous optimization with multiple constraints. In *Advances in Neural Information Processing Systems*.

Guillaume Lample, Sandeep Subramanian, Eric Smith, Ludovic Denoyer, Marc'Aurelio Ranzato, and Y-Lan Boureau. 2019. Multiple-attribute text rewriting. In *Proc. ICLR*.

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

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

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

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, Online. Association for Computational Linguistics.

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

Alisa Liu, Maarten Sap, Ximing Lu, Swabha Swayamdipta, Chandra Bhagavatula, Noah A. Smith, and Yejin Choi. 2021b. 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. Association for Computational Linguistics.

Guangyi Liu, Zichao Yang, Tianhua Tao, Xiaodan Liang, Junwei Bao, Zhen Li, Xiaodong He, Shuguang Cui, and Zhiting Hu. 2022. Don't take it literally: An edit-invariant sequence loss for text generation. In *Proc. NAACL*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach.

Ximing Lu, Sean Welleck, Peter West, Liwei Jiang, Jungo Kasai, Daniel Khashabi, Ronan Le Bras, Lianhui Qin, Youngjae Yu, Rowan Zellers, Noah A. Smith, and Yejin Choi. 2021a. Neurologic a*esque decoding: Constrained text generation with lookahead heuristics. *ArXiv*, abs/2112.08726.

Ximing Lu, Peter West, Rowan Zellers, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021b. NeuroLogic decoding: (un)supervised neural text generation with predicate logic constraints. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4288–4299, Online. Association for Computational Linguistics.

Eric Luhman and Troy Luhman. 2021. Knowledge distillation in iterative generative models for improved sampling speed.

Xinyao Ma, Maarten Sap, Hannah Rashkin, and Yejin Choi. 2020. PowerTransformer: Unsupervised controllable revision for biased language correction. In *Proc. EMNLP*.

Clara Meister, Tiago Pimentel, Gian Wiher, and Ryan Cotterell. 2022. Typical decoding for natural language generation.

Sewon Min, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2021. Noisy channel language model prompting for few-shot text classification. *arXiv preprint*.

Fatemehsadat Mireshghallah, Kartik Goyal, and Taylor Berg-Kirkpatrick. 2022. Mix and match: Learning-free controllable text generation using energy language models.

Artidoro Pagnoni, Vidhisha Balachandran, and Yulia Tsvetkov. 2021. Understanding factuality in abstractive summarization with FRANK: A benchmark for factuality metrics. In *Proc. NAACL*.

Biswajit Paria, Chih-Kuan Yeh, Ian E. H. Yen, Ning Xu, Pradeep Ravikumar, and Barnabás Póczos. 2020. Minimizing FLOPs to learn efficient sparse representations. In *Proc. ICLR*.

G. Parisi. 1981. Correlation functions and computer simulations. *Nuclear Physics B*.

Damian Pascual, Beni Egressy, Clara Meister, Ryan Cotterell, and Roger Wattenhofer. 2021. A plug-and-play method for controlled text generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3973–3997, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sam Patterson and Yee Whye Teh. 2013. Stochastic gradient riemannian langevin dynamics on the probability simplex. In *NuerIPS*.

John Platt and Alan Barr. 1988. Constrained differential optimization. In *Proc. NeurIPS*. American Institute of Physics.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1314–1324, New Orleans, Louisiana. Association for Computational Linguistics.

Shrimai Prabhumoye, Yulia Tsvetkov, Ruslan Salakhutdinov, and Alan W Black. 2018. Style transfer through back-translation. In *Proc. ACL*.

Jing Qian, Li Dong, Yelong Shen, Furu Wei, and Weizhu Chen. 2022. Controllable natural language generation with contrastive prefixes. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2912–2924, Dublin, Ireland. Association for Computational Linguistics.

Lianhui Qin, Vered Shwartz, Peter West, Chandra Bhagavatula, Jena D. Hwang, Ronan Le Bras, Antoine Bosselut, and Yejin Choi. 2020. 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. Association for Computational Linguistics.

Lianhui Qin, Sean Welleck, Daniel Khashabi, and Yejin Choi. 2022. COLD decoding: Energy-based constrained text generation with langevin dynamics.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *JMLR*.

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

Sravana Reddy and Kevin Knight. 2016. Obfuscating gender in social media writing. In *Proc. Workshop on NLP and Computational Social Science*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Emily Sheng, Kai-Wei Chang, Premkumar Natarajan, and Nanyun Peng. 2019. The woman worked as a babysitter: On biases in language generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3407–3412, Hong Kong, China. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

A. Sokal. 1997. *Monte Carlo Methods in Statistical Mechanics: Foundations and New Algorithms*.

Congzheng Song, Alexander Rush, and Vitaly Shmatikov. 2020. Adversarial semantic collisions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4198–4210, Online. Association for Computational Linguistics.

Yang Song and Stefano Ermon. 2019. Generative modeling by estimating gradients of the data distribution. In *Proc. NeurIPS*.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing NLP. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2153–2162, Hong Kong, China. Association for Computational Linguistics.

Eric Wallace, Mitchell Stern, and Dawn Song. 2020. Imitation attacks and defenses for black-box machine translation systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5531–5546, Online. Association for Computational Linguistics.

Minghan Wang, Guo Jiaxin, Yuxia Wang, Yimeng Chen, Su Chang, Hengchao Shang, Min Zhang, Shimin Tao,

and Hao Yang. 2021. How length prediction influence the performance of non-autoregressive translation? In *Proceedings of the Fourth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 205–213, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2020. Neural text generation with unlikelihood training. In *Proc. ICLR*.

Max Welling and Yee Whye Teh. 2011. Bayesian learning via stochastic gradient langevin dynamics. In *Proc. ICML*.

Kevin Yang and Dan Klein. 2021a. 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. Association for Computational Linguistics.

Kevin Yang and Dan Klein. 2021b. FUDGE: Controlled text generation with future discriminators. In *Proc. NAACL*.

Kexin Yang, Dayiheng Liu, Wenqiang Lei, Baosong Yang, Mingfeng Xue, Boxing Chen, and Jun Xie. 2022. Tailor: A prompt-based approach to attribute-based controlled text generation.

Dani Yogatama, Chris Dyer, Wang Ling, and Phil Blunsom. 2017. Generative and discriminative text classification with recurrent neural networks.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. SeqGAN: Sequence generative adversarial nets with policy gradient. In *Proc. AAAI*.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In *Proc. NeurIPS*.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2020. Fine-tuning language models from human preferences.

A MuCoLa Decoding Algorithm

We provide a formal algorithm for MuCoLa in 1.

Algorithm 1: MuCoLa: detailed decoding algorithm

Input: input sequence $\mathbf{x}$, output length L, base LM, attribute functions f_i and their respective thresholds ϵ_i, step sizes η, η_{max} (and schedule), η_λ, initial noise variance β_{init} (and schedule);

Result: output sequence $\mathbf{y}$

For all $n \in \{1, \ldots, L\}$, initialize $\tilde{e}_n^{(0)}$;

For all $i \in \{1, \ldots u\}$, initialize $\lambda_i^{(0)}$ as 0;

Initialize $\beta^{(0)}$ as β_{init};

Initialize $\eta^{(0)}$ as η;

for $t = 1, \ldots,$ MaxSteps **do**

  `// forward pass`

  compute the energy function $\mathcal{E}$ (see §3.3);

  `// backward pass`

  for all n, i, compute $\nabla_{\tilde{e}_n}^{(t-1)} = \frac{\partial \mathcal{E}}{\partial \tilde{\mathbf{e}}_n}$, $\nabla_{\lambda_i}^{(t-1)} = \frac{\partial \mathcal{E}}{\partial \lambda_i}$;

  `// Update the parameters`

  Sample $z^{(t-1)} \sim \mathcal{N}(0, I_d)$;

  Update $\tilde{\mathbf{e}}_{\mathbf{y}}^t = \text{Proj}_{\mathbf{E}}(\tilde{\mathbf{e}}_{\mathbf{y}}^{(t-1)} - \eta \nabla_{\tilde{\mathbf{e}}_{\mathbf{y}}}^{(t-1)} \mathcal{E} + \sqrt{2\eta^{(t-1)}\beta} z^{(t-1)})$;

  Update $\lambda_i^t = \max(0, \lambda_i^{t-1} + \eta_2 \nabla_{\lambda_i}^{(t-1)} \mathcal{E})$;

  update $\beta^{(t)}, \eta^{(t)}$ following the threshold update schedule.

end

Convert $\tilde{\mathbf{e}}^{(t)}$ to discete tokens $\hat{\mathbf{y}}^{(t)}$ by nearest neighbor search.;

return $\arg\min_t \{-\log P(\hat{\mathbf{y}}^{(t)}|\mathbf{x}) : \forall i, f_i(\tilde{\mathbf{y}}^{(t)}|[\mathbf{x}]) \leq \epsilon_i\}$;

B Related Work

Controllable Text Generation Prior work in this area can be divided into three categories: The first focuses on training models with specific control codes via pretraining (Keskar et al., 2019) or finetuning (Gururangan et al., 2020; Chan et al., 2021) for prompt based generation and generative models for tasks such for style transfer (Lample et al., 2019; Ziegler et al., 2020; Prabhumoye et al., 2018; Yu et al., 2017). These methods are naturally difficult to extend to new controls as it requires retraining the models.

The second category includes decoding approaches from LMs without modifying them (MUCOLA falls under this category). Most prior work in this space has explored methods to modify left-to-right search or sampling algorithms by modifying the output probability distribution at each step using different control functions. Dathathri et al. (2020); Krause et al. (2020a); Yang and Klein (2021b); Liu et al. (2021b) apply this approach for soft constraints defined by classifiers and LMs whereas Lu et al. (2021b,a); Pascual et al. (2021) develop heuristic control functions for keyword based constraints. In contrast, we show that MUCOLA is able to incorporate both kinds of constraints. Since these approaches generate one token at time and do not allow modifying a token once it is generated, they are not ideal for controls that are conceptually defined on the entire sequence. Hence, prior work has also explored non-autoregressive decoding methods (Mireshghallah et al., 2022). Most closely related to MUCOLA is Kumar et al. (2021) which propose a gradient-based decoding algorithm which we extend to generate multiple samples. Also related is Qin et al. (2022) that perform Langevin Dynamics in the simplex space to incorporate control by representing the energy function as a linear combination of control functions. In contrast, we represent the energy functions as a Lagrangian and perform these updates on a much smaller embedding space allowing us to generate longer sequences.

The third category includes more recent few-shot methods which rely on prompting large LMs such as GPT3 to incorporate controls based on demonstrations (Qian et al., 2022; Yang et al., 2022; Carlsson et al., 2022). MUCOLA is an orthogonal approach to this work and can be applied on top of prompt-based solutions to increase control satisfaction.

While this work and the work described above focuses on controlling attributes of individual text outputs, in related work, Khalifa et al. (2021); Korbak et al. (2022) develop approaches for distributional control of various attributes in generated text corpora.

Gradient-based Sampling Langevin Dynamics and other gradient-based MCMC methods have been developed for generative modeling in continuous domains such as images (Song and Ermon, 2019) and audio (Jayaram and Thickstun, 2021) among others where the models are trained to predict the gradients (via a score function) directly whereas MUCOLA requires a backward pass to compute them. Also related are diffusion models which have obtained state-of-the-art performance for many generative tasks (Ramesh et al., 2022; Ho et al., 2022). Similar ideas have also been applied to train text generation models in concurrent work with promising results for incorporating controls (Li et al., 2022).

C Implementation Details

Here, we describe additional implementation details as part of the experimental setup described in §4

Noise Schedule The amount of noise in each update is controlled by β (§3.2) which represents the variance of the noise term. We initialize β with 5.0 and decrease it to 0.05 in a geometric progression for 100 steps after which we keep it constant at 0.05 for the remaining 150 steps. The range of β is guided by best practices in Song and Ermon (2019) prescribing the initial variance to be close to the maximum distance between any two vectors in the input space and the minimum value being close to 0. This schedule allows for sufficient exploration in the beginning helping in diversity, while, leaving enough iterations for optimizing the final output into a fluent sequence.

Step Size and Selection Criterion The step-size η in projected gradient descent depends on the geometry of the embedding space of the underlying language model. Since we project back the update at every step to $\mathbf{E}$, if the update term is not big enough in the projected gradient update, the sequence at step $t+1$ would remain the same. This observation provides a very simple criterion for early stopping and selecting the best output out of all iterations. When the additive noise is small

(near the end of optimization), the update term can be small due to following factors: (a) η is small, (b) the gradient $\nabla \mathcal{E}_{\tilde{\mathbf{e}}}$ is small which implies the output sequence has "converged". Hence, we define a schedule on the step-size as follows: we start with a step-size η, and update the outputs using Langevin Dynamics until the sequence stops updating, i.e., the update value becomes too small (and satisfies all constraints). Now, to make sure that this is a convergence point and not a result of the step size being too small, we update the step size linearly to η_{max} in s steps[11]. If the sequence does not update in s steps, we stop early and predict the output. Otherwise, the process continues. If it does not stop early at the end of maximum number of steps, we predict the output with the highest likelihood which repeated at least 5 times. In the event, no such repetition is observed, we deem the optimization as "failed" and restart. If the restarts also fail, we just predict the autoregressive output (which in our experiments is obtained with nucleus sampling with $p = 0.96$). This fallback mechanism ensures that the output, irrespective of the constraint satisfaction is always a sample of P while preventing generating half-baked outputs.

Multipliers Update Schedule We initialize each of the multipliers λ_i with 0, update the multipliers via gradient ascent every 20 steps using the step-size 1.0. In addition, if the sequence stops updating at a certain iteration (as described above) and i-th constraint is not satisfied, we update λ_i at every iteration till the sequence starts updating again. This schedule prevents fluctuation in the multiplier values when the noise is high in the early iterations and the sequence has not converged to anything fluent while still allowing updates when required (Platt and Barr, 1988; Paria et al., 2020).

D Training Details for Soft Constraint Models

Since we decode by computing gradients over token embeddings, it requires that all constraint models share the same embedding table $\mathbf{E}$ as that of the underlying language model P. Since any typical text based model involves an embedding table, we can train a constraint using such a model by simply initializing its embedding table with $\mathbf{E}$. In principle, this initialization allows using any off-the-shelf pretrained model as a constraint function by finetuning it on appropriate data. In our experiments, we use the following models in different experiments:

Toxicity Classifier For toxicity avoidance (§5.1), we finetune roberta-base (Liu et al., 2019) with a binary classification head using a dataset of human-annotated comments from the Jigsaw Unintended Bias In Toxicity Classification Kaggle Challenge. The dataset has $\sim 160K$ toxic comments and $\sim 1.4M$ non-toxic comments. We first balance this dataset by subsampling $160K$ examples from the non-toxic class. We replace the embedding table of roberta-base with that of the underlying LM (GPT2-Large in our case). To address the dimension mismatch of the two embedding tables, during finetuning, we also learn a linear projection matrix which transforms base LM embedding to a smaller dimension of roberta-base. We keep base LM embedding frozen during this finetuning. We use a learning rate of $1e-5$ and train for 3 epochs with an effective batch size of 64. We choose a checkpoint with an accuracy of $\sim 93\%$ on a heldout development set.

Sentiment Classifiers For sentiment control experiments in §5.2, we experiment with different kinds of constraints defined using both classifiers and language models. For both setups we use two datasets: SST-2 corpus (Socher et al., 2013) containing $\sim$4K examples in Movie reviews for each class; and Yelp polarity corpus containing $\sim$280K examples for each class containing a mixed domain of reviews.

For discriminative classifiers, we also finetune roberta-base using the same setup and hyperparameters as the toxicity classifier. Our best model obtains an accuracy of $\sim 92\%$ on the SST-2 test set and $\sim 98\%$ on the Yelp test set.

To train the generative classifiers, we finetune GPT2-Large (and do not need to substitute any embedding tables) keeping the embedding table frozen. We use the loss $-\log p_{\text{gen}}(\text{label}|x)$ for each training instance where $p_{\text{gen}}(\text{label} = 0|\text{text}) = p_{\text{LM}}(\text{text}|\text{label} = 0)/(p_{\text{LM}}(\text{text}|\text{label} = 0) + p_{\text{LM}}(\text{text}|\text{label} = 1))$. This is due to Bayes' rule ($p(\text{label})$ vanishes as we set it to 0.5 for balanced datasets). Here $p_{\text{LM}}(\text{text}|\text{label})$ is obtained using the language model by computing the probability of the text conditioned on the input token "positive" for the positive label and "negative" otherwise.

We again follow the same training hyperparam-

[11] s is empirically defined as 40 in all our experiments.

eters for this setup. On SST-2 test set, we obtain an accuracy of $\sim$95% and on Yelp, we obtain an accuracy of $\sim$98%.

E Additional Explanation and Results for Hard Constraint

The keyword distance function $d(w, \tilde{\mathbf{e}})$ is computed in three steps. First, we convert each $\tilde{e}_n$ to a "probability" over the vocabulary as,

$$\pi_n = \text{softmax}(-\|\tilde{e}_n - e_1\|_2^2, \ldots, -\|\tilde{e}_n - e_{|\mathcal{V}|}\|_2^2)$$

where $\{e_1, \ldots, e_{|\mathcal{V}|}\}$ are entries in the embedding table $\mathbf{E}$. Since each $\tilde{e}_n$ itself also corresponds to a vector in $\mathbf{E}$, if n-th token in the sequence is w, then, $\|\tilde{e}_n - e_w\|_2^2$ would be 0 leading to $\pi_{n,w} = \max_j \pi_{n,j}$. That is, maximizing $g_n = \log \pi_{n,w}$ with respect to $\tilde{e}_n$ would nudge it towards the e_w. Since, we don't know which index we want w to appear at in advance, following (Liu et al., 2022; Qin et al., 2022), we (soft) sample it using $\pi_{n,w}$ as weights. This brings us to the second step, as we define, $q = \text{GUMBEL-SOFTMAX}(-g_1/\tau, \ldots, -g_N/\tau)$ where τ is the temperature. We use hard sampling here to ensure q is one-hot. Finally, we define the constraint function as, $d(w, \tilde{\mathbf{e}}) = \sum_{i=n}^{N} -q_n g_n$. Intuitively, this function aims to generate the word w wherever it already has a high chance of getting generated (measured via $\pi_{n,w}$'s). Stochasticity in this function allows for exploration. This function can be easily extended from words to phrases of length l, $w = (w_1, \ldots, w_l)$ by defining $g_n = \frac{1}{l}\sum_{u=1}^{l} -\log \pi_{w_u, n+u}$. This computation can be efficiently done on a GPU using a convolution operation (Liu et al., 2022).

Based on this definition, we define the keyword constraint for MUCOLA as $d(w, \tilde{\mathbf{e}}) \leq -\log \pi_{w,w} + \delta$, where δ is a small positive value (we set it as 0.1). π_w is a slight abuse of notation to define a distribution similar to π_n (n refers in an index in sequence whereas w refers to an index in $\mathcal{V}$). Note that the threshold for each keyword is different.[12]

Intuitively, if w appears in the output at the k-th position, then $\pi_{k,w} = \pi_{w,w} = \max_j \pi_{k,j}$ with q_k as 1. This reduces the distance function to $-\log \pi_{k,w}$ which is less than the defined threshold. Conversely, if w does not appear in the output, for each n, $-\log \pi_{n,w}$ would be higher than $\log \pi_{w,w}$ and the constraint remains unsatisfied. This is due to an empirical observation we make in all embedding tables we use, that $\pi_{w,w} = \max_j \pi_{w,j} = \max_j \pi_{j,w}$. In other words, not only is the probability of a word under its own distribution pi_w the greater than probability of all other words (since the corresponding distance is 0), it is also larger than w's probability under all other distributions defined for any word in the vocabulary. Under the assumption that minimum distance between any two vectors in the table is greater than a small positive value, we conjecture this claim to be true for any embedding table.

Open-Ended Keyword Guided Generation In addition to COMMONGEN, we report results on ROC (Pascual et al., 2021) task where given 5 keywords, the goal is generate a sequence of max length 90 containing those terms. For both datasets, for set of keywords, we generate samples of length 10, 20, and 40 (with 3 restarts for each) and after all iterations are complete, we continue generating more tokens autoregressively until a maximum of 40 (90 in case of ROC) tokens are generated or end of sequence token is generated. Finally, we evaluate on one output which satisfies the constraints and has the lowest perplexity according to the LM. We compare MUCOLA with the best reported results in (Qin et al., 2022) and (Pascual et al., 2021) and corresponding baselines. The results for ROC can be found in table 9.

Terminology Constrained Translation We follow the setup in Dinu et al. (2019) and use an off-the-shelf English to German translation model by MarianMT (Junczys-Dowmunt et al., 2018) to translate a subset of WMT17 en-de test set (Bojar et al., 2017). The constraint here is to integrate a given custom terminology into the translation output; where the terms are automatically created from the IATE EU terminology database for 414 test sentences (with 1 to 3 terminology constraint per example). We use Lu et al. (2021a) as our best baseline and also report other baselines reported by them. We generate each translation by first generating with beam search unconstrained (with beam size of 6). If this output is of length L. We use MUCOLA to generate sequences of

[12] While we do not experiment with it in this work, the constraint $K(w, \tilde{\mathbf{e}})$ can be easily extended to setup where at least one out of n given words (for example different surface forms of the same root), $S = \{w_1, \ldots, w_p\}$ must appear in the output by defining a new constraint as $K(S, \tilde{\mathbf{e}}) = \max_{w_i \in S} K(w_i, \tilde{\mathbf{e}})$ or its soft version using the gumbel-softmax trick.

Method	BLEU	Coverage
Unconstrained	32.9	85.3
Post and Vilar (2018)	33.0	94.3
Neurologic*	33.5	97.2
MUCOLA	33.1	100

Table 4: Results for terminology constrained en–de translatoin (§E)

length $\{L, L+1, \ldots, L+10\}$ and select the generation which has the highest length-normalized log-probability as the final translation. We evaluate on BLEU score[13] and coverage accuracy. As reported in table 4, MUCOLA obtains perfect (100%) coverage while at the same maintaining BLEU score.

Entity Constrained Summarization In this setup, we do a preliminary exploration on text summarization with a constraint that a specific entity must appear in the summary given the article. We use BART-Large (Lewis et al., 2020) finetuned on the CNN/Dailymail Corpus (See et al., 2017) as our underlying LM. First, we obtain all named entities appearing in the article using an off-the-shelf recognizer[14]. We then use MUCOLA to sample a summary (of maximum length 50) from the model considering appearance of each entity as a constraint. We show selected examples with promising results in table 16, table 17 and table 18. Evaluating this setup is non-trivial, since it adds new sentences/phrases to the summary and will naturally perform poorly on standard reference based metrics such as ROUGE. Hence, we leave this evaluation for future work.

F Additional Results for Soft Constraints

Toxicity Avoidance For human evaluation, we follow an A/B testing framework and compare MUCOLA and DExperts. We sample 200 prompts from the test set and consider 2 generations per prompt. We ask each annotator to rank the outputs from the two approaches on (1) toxicity if one output is more or less toxic than the other, or if both are equally toxic/non-toxic, (2) topicality: is the generation coherent with the prompt and follows the same general topic, and (3) fluency: if the outputs have any grammatical mistakes. We collect 3 annotations per pair. We find that in terms of toxicity, both models perform similarly with an average 8.5% annotations preferring MUCOLA's outputs compared to 9.5% for DExperts (rest are equally ranked). On topicality, 22.5% of annotations prefer MUCOLA's outputs while 19% prefer Dexperts (rest are equally ranked). On fluency, both models perform similarly with 22.5% and 23% in each method's favor and rest equally ranked.

Sentiment Control We present the full set of results for sentiment control experiments in tables 5, 6, 7 and More details can be found in the captions. For human evaluation, we similarly follow an A/B testing framework and compare MUCOLA and DExperts (for outputs of length 20). We consider all 15 prompts from the test set and consider 2 generations per prompt. We ask each annotator to rank the outputs from the two approaches on (1) positivity if one output is positive and the other is not, or if both are positive/not-positive, (2) topicality: is the generation coherent with the prompt and follows the same general topic, and (3) fluency: if the outputs have any grammatical mistakes. We collect 3 annotations per pair. We find that in terms of positivity, on an average 23.3% annotations prefer MUCOLA's outputs compared to 16.7% for DExperts (rest are equally ranked). On topicality, 26.7% of annotations prefer MUCOLA's outputs while 13.3% prefer Dexperts (rest are equally ranked). On fluency, MUCOLA slightly underperforms with 7.8% and 10% in each method's favor and rest equally ranked.

G Example

We provide selected examples from each of our experiments in tables 11, 12, 13, 14, 15 and 16.

H Additional Discussion and Analysis

Speed and Memory Requirements Generating a sequence of length L using MUCOLA requires maintaining $L \times d$ parameters. In contrast, performing Langevin Dynamics in the vocabulary space requires $L \times |\mathcal{V}|$ parameters ($|\mathcal{V}| >> d$). In this analysis, we empirically verify the benefits of our setup. Taking GPT2-Large as the underlying LM (with 774M parameters), and three commercially available GPUs with different RAM sizes commonly used in academic settings–Nvidia GeForce RTX 2080 Ti (12GB), GeForce RTX 3090 Ti (24GB) and RTX A6000 (48GB)–we decode using our approach with token embeddings and an

[13] For fair comparison, we compute a tokenized BLEU score reported by the baselines following https://github.com/INK-USC/CommonGen/tree/master/evaluation

[14] https://huggingface.co/dslim/bert-base-NER-uncased

Approach	Setting	% Positive Sentiment (↓)			Fluency		Diversity		
		C1	C2	C3	Perplexity	CoLa Accuracy	Dist-1	Dist-2	Dist-3
GPT-2	-	49.0	45.0	62.0	54.9	68.7	0.66	0.87	0.81
DAPT	SST-2	71.3	66.7	75.0	98.0	64.0	0.64	0.85	0.79
DAPT	Yelp	64.0	71.3	79.7	146.6	58.0	0.60	0.84	0.80
FUDGE	SST-2	71.7	70.0	79.0	11.4	82.7	0.53	0.76	0.77
FUDGE	Yelp	71.7	73.7	84.7	11.8	85.7	0.53	0.76	0.77
MUCOLA-DISC	SST-2	90.0	81.7	93.3	28.8	67.3	0.52	0.73	0.74
MUCOLA-DISC	Yelp	88.3	87.0	91.7	32.9	64.3	0.52	0.74	0.75
MUCOLA-TWO-DISC	Yelp, SST2	94.0	91.3	94.7	29.4	55.0	0.46	0.68	0.71
GEDI	SST-2	99.7	91.0	99.3	625.7	54.3	0.65	0.76	0.71
GEDI	Yelp	82.0	90.0	89.0	444.9	40.0	0.71	0.78	0.66
MUCOLA-GEN	SST-2	91.3	88.3	97.0	57.2	68.0	0.50	0.69	0.70
MUCOLA-GEN	Yelp	86.3	89.7	91.7	53.0	67.7	0.50	0.70	0.70
MUCOLA-PROMPT	-	89.0	88.7	94.7	43.7	66.7	0.49	0.72	0.73
DEXPERTS	SST-2	93.1	86.9	94.9	75.2	71.5	0.63	0.85	0.81
DEXPERTS	Yelp	80.3	88.5	88.8	116.3	67.5	0.67	0.84	0.79
MUCOLA-DEXPERTS	SST-2	93.0	88.0	94.0	41.4	66.3	0.47	0.71	0.73
MUCOLA-DEXPERTS	Yelp	74.3	74.0	83.3	72.5	66.0	0.52	0.73	0.74

Table 5: Positive sentiment control results on outputs of length 12. For each baseline (FUDGE, GEDI and DEXPERTS), we convert their respective constraints to a classifier (generative or discriminative; see §5.2). For FUDGE and GEDI, we show improvements on both control (% positive sentiment) and fluency (Perplexity) without any model specific changes. This improvement is consistent on models trained on both datasets (SST-2 and Yelp). DEXPERTS outperforms all baselines here including our method.

Approach	Setting	% Positive Sentiment (↑)			Fluency		Diversity		
		C1	C2	C3	Perplexity	CoLa Accuracy	Dist-1	Dist-2	Dist-3
GPT-2	-	46.7	47.7	61.3	38.6	78.7	0.64	0.90	0.88
DAPT	SST-2	73.6	70.0	78.3	76.9	70.7	0.64	0.89	0.86
DAPT	Yelp	65.0	75.0	80.7	86.6	69.7	0.59	0.88	0.87
FUDGE	SST-2	67.6	63.0	79.3	10.3	94.0	0.51	0.80	0.84
FUDGE	Yelp	71.0	70.0	79.3	10.6	89.0	0.53	0.81	0.85
MUCOLA-DISC	SST-2	84.6	77.5	88.0	27.9	80.8	0.50	0.81	0.82
MUCOLA-DISC	Yelp	83.0	83.6	83.0	**32.2**	76.0	0.50	0.75	0.80
MUCOLA-TWO-DISC	Yelp, SST2	**93.7**	**91.0**	**96.0**	28.9	76.7	0.53	0.77	0.74
GEDI	SST-2	99.0	96.3	99.7	268.7	54.0	0.69	0.87	0.84
GEDI	Yelp	84.0	95.7	91.0	208.3	44.0	0.76	0.87	0.81
MUCOLA-GEN	SST-2	86.3	80.3	93.3	45.6	77.7	0.50	0.74	0.78
MUCOLA-GEN	Yelp	79.7	83.0	90.0	27.2	72.3	0.50	0.82	0.86
MUCOLA-PROMPT	-	87.3	91.0	93.0	53.0	77.2	0.54	0.82	0.80
DEXPERTS	SST-2	91.2	83.4	95.4	55.37	81.6	0.61	0.89	0.87
DEXPERTS	Yelp	81.1	85.8	92.5	95.87	71.7	0.66	0.89	0.87
MUCOLA-DEXPERTS	SST-2	89.3	83.7	93.7	**32.2**	79.7	0.51	0.78	0.80
MUCOLA-DEXPERTS	Yelp	78.0	75.7	83.3	34.1	68.3	0.52	0.77	0.81

Table 6: Positive sentiment control results on outputs of length 20. For each baseline (FUDGE, GEDI and DEXPERTS), we convert their respective constraints to a classifier (generative or discriminative; see §5.2). For FUDGE and GEDI, we show improvements on both control (%positive sentiment) and fluency (Perplexity) without any model specific changes. This improvement is consistent on models trained on both datasets (SST-2 and Yelp).

Approach	Setting	% Positive Sentiment (↓)			Fluency		Diversity		
		C1	C2	C3	Perplexity	CoLa Accuracy	Dist-1	Dist-2	Dist-3
GPT-2	-	47.7	44.3	61.3	36.3	78.3	0.59	0.92	0.94
DAPT	SST-2	93.0	84.3	91.7	55.3	88.0	0.61	0.92	0.94
DAPT	Yelp	72.3	80.7	85.0	46.1	84.3	0.51	0.90	0.94
FUDGE	SST-2	71.0	61.3	84.7	8.5	98.3	0.47	0.83	0.92
FUDGE	Yelp	72.3	68.0	80.3	8.3	99.0	0.47	0.83	0.92
MUCOLA-DISC	SST-2	88.7	81.0	91.3	15.3	72.7	0.42	0.68	0.76
MUCOLA-DISC	Yelp	70.7	74.3	81.3	19.1	77.7	0.48	0.77	0.85
MUCOLA-TWO-DISC	Yelp, SST2	94.0	91.3	94.7	29.4	75.0	0.57	0.78	0.79
GEDI	SST-2	86.7	98.7	96.7	148.4	68.3	0.75	0.94	0.93
GEDI	Yelp	99.7	98.7	100.0	114.5	74.3	0.66	0.93	0.93
MUCOLA-GEN	SST-2	85.0	76.3	91.0	22.5	63.7	0.44	0.71	0.78
MUCOLA-GEN	Yelp	77.7	80.7	88.3	23.4	65.0	0.43	0.69	0.76
MUCOLA-PROMPT	-	81.3	83.0	92.7	18.2	72.0	0.39	0.67	0.77
DEXPERTS	SST-2	98.1	92.0	99.5	39.5	88.5	0.57	0.91	0.94
DEXPERTS	Yelp	87.2	91.7	94.9	54.0	77.3	0.62	0.92	0.93
MUCOLA-DEXPERTS	SST-2	72.7	71.7	84.7	28.2	69.0	0.45	0.75	0.83
MUCOLA-DEXPERTS	Yelp	62.3	61.7	75.7	18.8	81.0	0.48	0.77	0.83

Table 7: Positive sentiment control results on outputs of length 50. For each baseline (FUDGE, GEDI and DEXPERTS), we convert their respective constraints to a classifier (generative or discriminative; see §5.2). For FUDGE and GEDI, we show improvements on both control (% positive sentiment) and fluency (Perplexity) without any model specific changes. This improvement is consistent on models trained on both datasets (SST-2 and Yelp).

Threshold	Initialization	Toxicity		Fluency		Diversity		
		Avg. Max. Toxicity	Toxicity Prob	PPL	CoLa Accuracy	dist-1	dist-2	dist-3
0.5	Random	0.351	0.268	32.1	87.5%	0.58	0.85	0.85
0.3	Random	0.352	0.200	33.0	87.5%	0.58	0.85	0.85
0.1	Random	0.320	0.158	31.2	86.3%	0.56	0.83	0.83
0.01	Random	0.302	0.094	28.8	87.1%	0.55	0.82	0.83
0.01	Zeros	0.302	0.094	35.3	85.8%	0.55	0.81	0.82
0.01	Greedy	0.302	0.115	28.6	86.6%	0.55	0.81	0.83

Table 8: Ablations on Toxicity Avoidance showing the effect of changing classifier threshold (ϵ) on toxicity metrics, and initialization on diversity metrics. Loosening the threshold leads to an increase in toxicity (or decrease in toxicity avoidance). Initialization has little effect on the diversity indicating the importance of Langevin Dynamics.

	Coverage (%)	Fluency (PPL)	Repetition Rate
Plan-and-Write	96	33.9	25.7
CGMH	97	127.8	1.6
GPT-2 fine-tuned	72	89.4	1.8
GPT-2+K2T	**100**	48.8	1.5
MUCOLA	100	**29.4**	0.5

Table 9: Results of lexically constrained decoding on the ROC dataset (with 5 keyword constraints). We decode with MUCOLA with lengths 10, 20 and 40, and if the constraint is satisfied we continue generating autoregressively for 90 tokens using nucleus sampling ($p = 0.96$).

Constraint (#Parameters)	12GB GPU		24GB GPU		48GB GPU	
	Embeds	**Simplex**	**Embeds**	**Simplex**	**Embeds**	**Simplex**
None (0)	200	Failed	1000	20	1000	50
One Classifier (125M)	200	Failed	500	20	1000	50
Two Classifier (250M)	50	Failed	500	20	1000	50
One LM (747M)	Failed	Failed	500	20	200	20
Two LMs (1.4B)	Failed	Failed	200	10	500	20

Table 10: Maximum sequence length that can be generated with GPT2-Large (747M parameters) for different commonly used GPUs with MuCoLa (with embeddings) and with an ablation using vocabulary sized representations (referred to as Simplex). Failed denotes out of memory (OOM) error.

Prompt	Republican Presidential candidate Donald Trump weighed in on Paris attacks saying, "sneaky,
GPT2	weak and stupid" Muslims, the press wants to use the Paris attacks to their advantage. "Okay,
GeDi	corrupt political leaders" had helped promote Radical Islam. Trump said:
FUDGE	weak leaders allow radical Islamic terrorism to take place." Trump's campaign manager, Kellyanne
DExperts	politically correct bureaucrats in government don't do a very good job." Interested in France? Add
MuCoLa-DISC	very sneaky" Obama from Chicago goes around telling others what to do. I'm putting out a call for everyone
Prompt	"I expected him to carry on [in] the article saying that I was
GPT2	a stupid idiot," he says. But ultimately the material he wrote isn't about West. It's
GeDi	involved, until we find something else.\n\n'Then he got out of the car and attacked
FUDGE	not a racist, but instead that I am a 'racist'. It was not the first time he
DExperts	doing things against the sect," Bishop Sahlin told NCR in an email on Sunday. "But
MuCoLa-DISC	a woman, but he didn't," the woman said. "I was a little bit surprised,"

Table 11: Selected examples from Toxicity Avoidance Experiment (§5.1) generated by different methods.

Prompt	Once upon a time
GPT2	, you had to fire the cannon at sea at noon when
GeDi	a young priest traveling the world taught the world the meaning of
FUDGE	, in a land far away, there lived a man with
DExperts	, white women ruled both Australia and America and cherished his nation
MuCoLa-DISC (SST2)	, the people of the United States were a people of the
MuCoLa-DISC (Yelp)	, I was a great big-time, all-American
MuCoLa-TWO-DISC	, the people of the world were a very different and powerful
MuCoLa-PROMPT	you start with just Bluetooth and now with this versatile module you

Table 12: Examples of length 12 by the prompt "Once upon a time" generated by different methods.

Prompt	Once upon a time
GPT2	, you had to fire the cannon at sea at noon when all the other sailing vessels were under way
GeDi	unseen world through vivid mystical experience! One enjoys becoming connected with the unseen. Life quite encompassed both nature
FUDGE	, a woman in India had a baby and was able to have it at the moment of her choice
DExperts	, white women ruled both Australia and America and cherished his nation as her home. Her words resonate with
MUCOLA-DISC (SST2)	, the world was a very beautiful, and a very good, place. The people were kind and
MUCOLA-DISC (Yelp)	, I had a great time. I was a very nice and very good-looking man. I
MUCOLA-TWO-DISC	, I enjoyed the wonderful family and friends I had in the community.\n\n I was a good
MUCOLA-PROMPT	, I was a nobody, but eventually I became one of the biggest names in the nation.\n

Table 13: Examples of length 20 given the prompt "Once upon a time" generated by different methods.

ablation with vocabulary sized representations (logits plus softmax). We generate sequences of length $\{10, 20, 50, 100, 200, 500, 1000\}$, and consider 5 constraint settings: (1) no constraint, (2) one classifier (same as §5.1 containing $\sim 125M$ parameters (3) two-classifiers (MUCOLA-TWO-DISC) with a total $\sim 250M$ paramaters (4) a LM based generative classifiers (same size as GPT2-Large), (5) and LM based generative classifier using two LMs (double the size of GPT2-Large). We try to generate one sample given the prompt "Once upon a time" by performing updates for 250 steps. We report the longest sequence that each setup is able to work with. The results are summarized in table 10. Overall, we see that much longer sequences can be generated with MUCOLA than the ablation. MUCOLA is comfortably able work with up to a 1000 tokens without constraints (and 200 with two large constraints with larger GPUs) while the ablation fails beyond 50 tokens (20 with constraints).

We present the result for ablations on sources of diversity (§7) in table 8.

Prompt	Once upon a time
GPT2	, you had to fire the cannon at sea at noon when all the other sailing vessels were under way. It has been a close quarter battle. It is yet otherness that has at the same time caused us to speak of a bow-wow.
GeDi	civilians lived alongside peaceful bystanders. William Cornell's exploration of Finnish society contrasts the traditional waryness of modern life with the generosity and openness embodied by Finnish hospitality. Transformed for centuries from refugees in wartime Russia, Finns welcomed their
FUDGE	, there was a man named John. He and his friend, Paul, were in a diner. They were in the middle of a conversation. Paul said to John, "John, I just want to make sure that you understand why we are having
DExperts	, white women ruled both Australia and America and cherished his nation as her home. Her words resonate with millions who lived through the trials of the last decade as Islam \u2013 still controversial today \u2013 entered Australia's first democratically elected Muslim prime minister and wounded Australia's
MUCOLA-DISC (SST2)	, I was a big fan of the "The Big Lebowski" and the "The Big Lebowski" was a big part of my life. I was a big fan of the "
MUCOLA-DISC (Yelp)	, the world was a very different place. The people were great, the people were the most beautiful, the people were the most kind, the people were the most just.\n\nAnd the world was a very interesting place. The people were great
MUCOLA-TWO-DISC	, the captain was a hero, a most important and a most powerful one. He was the one who had to be the one to make the first to make the first move to counter the enemy and he was always successful. The great and the mighty
MUCOLA-PROMPT	, I would have never believed that I could make sushi from a simple, but delicious, recipe. I have been making this for a while and it is a great, one-dish, a-day-for-a-sushi

Table 14: Examples of length 50 by the prompt "Once upon a time" generated by different methods.

Lexical Constraints	Length	Output
goal, player, shot, take	10	The **goal** of the **player shot** is to **take** a direct route to the net. The target line is a straight line between the two feet. The distance from the line-cutters to the goal line is
catch, dog, frisbee, throw	10	The first **dog** to **catch** a **throw** from the **frisbee** is usually a happy ending, right? WRITTEN BY: Laura S. Laura S. Laura S. Laura S. Laura S
bike, ride, trail, wood	10	The **bike ride trail** at the **trail**head is **wood**ed and has an easy grade (7%) that allows you to reach a destination while enjoying the views. As a general guide to trails in the
front, guitar, microphone, sit	20	The **microphone** is in **front** of the **sit**-down area and the **guitar** is in the back. The two are plugged into the mic's input jack. The sound can be recorded on video or recorded with
dog, leash, sidewalk, walk	20	The **leash walk** on the **sidewalk** is a great way to get to know your new **dog**. It is a great exercise and a way to take pictures of your new dog. Many people take photos with their dog
music, perform, routine, stage	20	The first **stage** of the **routine** is to have the person in the **music**- and **perform**-in-audition pose the questions to the computer. The computer then asks any number of questions in response to these
drill, field, run, team	40	The New York **field drill team** is **run** by the New York-based American Field and R.A.T. (A.F.R.T.) and is the **team**'s official military training facility. The **team**'s purpose is to help both
cook, food, pan, stove	40	I'm a big **food**ie fan. I **pan**-fry, I **cook stove**-top, I make a lot of my own own. (You had better come find me, or I'll get you!) And I've spent a fortune on
compete, field, game, team	40	The **team** is in a **field** of their own, and the only **field** they compete in is the one that is in their own head. I don't think that is a good **game** to be in
fabric, machine, piece, sew, stitch	10	The first **machine stitch sew**-on **fabric piece** is a **fabric** piece with a pattern edge facing up, with the top edges being 1/2 inch from the edge. As it rises you should cut
bean, bowl, machine, pour, roast	10	The **bean pour bowl roast** is a **machine** that is able to roast in the oven at high temperatures, it takes a large amount of heat (typically 900 F+) and will have a very small surface to
beach, dog, hold, jump, leash	10	The **jump leash** is great for **dog beach** for **hold** down the kennel, and its lightweight that you can see the dog to keep her out in the open and out of the water at the kennel. For
back, floor, lie, sit, talk	20	The first time I **sit** down to a **talk**, I **lie** on my **back** and I **floor** it. If I'm going to sit down to lecture, you need to lift me up and then you have
bowl, fall, grinder, meat, put	20	The **fall** of the **grinder** is a good thing. The **meat bowl** is not. I **put** the **meat bowl** back in my fridge to chill out, but by the time I was ready for dinner one morning
ball, fire, hold, juggle, light	20	The first time I **juggle ball**, I **hold** the **ball** in my left hand and **light** the ball with my right hand. I like to go up and down the center of my body, and then do it
front, listen, microphone, music, stand	40	I **listen** to **music**, and I **stand** in **front** of a **microphone**, and I do it. I don't have to have a microphone, and I don't have to do it. That's what's going
artist, audience, belt, fight, front	40	The first **belt**-and-cuff-wearing **artist** to **fight** in **front** of a live **audience** in the United States, the "B.A.P B-S-T" (Bitch, Asshole and Steroid) rapper went
give, instruction, machine, sew, use	40	The **machine** is very simple, but it is very very important. The more **instruction** you use, the more you can **sew**. The more you can do, the more you can **give**. The more efficient

Table 15: Examples of lexically constrained outputs generated by our model on the COMMONGEN dataset. Length refers to the original length of the sentence on which MUCOLA was performed. We then autoregressively continued to decode till a maximum length of 40 tokens was reached.

Arsenal defender Per Mertesacker has tipped compatriot Jurgen Klopp to make his mark in the Barclays Premier League if he opts to continue his career in England. Klopp, 47, announced earlier this week that he would end his seven-year stint at Borussia Dortmund when the current season draws to a close, prompting fresh speculation that he could head for the Premier League. Manchester City have already indicated that a man who has also been linked with Manchester United and Arsenal in the past, is not in their sights, but Germany international Mertesacker insists Klopp would be a good fit in the English top flight. Jurgen Klopp has revealed he will be vacating his role as Borussia Dortmund boss at the end of the season . Arsenal vice-captain Per Mertesacker says Klopp would be a top manager in the Premier League . Klopp chats with Dortmund defender Erik Durm during a training session in Dortmund on Wednesday . He said: 'I've got some nice experiences in the Premier League and of course it would be nice if a German coach would take the challenge of working in the Premier League. 'It's not so good for Dortmund that he is leaving but hopefully one day he will manage abroad. I think his passion would fit and to see him in England would be very interesting. 'Everyone has their philosophy and I think Jurgen Klopp has proved that he's top-level and can teach a lot.' However, Mertesacker insisted Klopp, whose side are 10th in the Bundesliga table, will need time to decide on his future after a largely successful spell in Dortmund which has brought two league titles and a Champions League final appearance. He said: 'I think he should just finish the season with Dortmund and then he should be given time. 'We'll see what he does next, but I think he's fought his way out of all situations and I think that this time he will find a path that gives him a new challenge. 'But firstly, I wish him all the best and time to think about his achievements. Sometimes you can underestimate what it's like going straight into a new job. I think you should give him time - and I wish him all the best.' Klopp waves to the fans after Dortmund's Champions League game against Arsenal in November . The German boss has enjoyed a huge amount of success at Dortmund and won the Bundesliga title twice . But for all that a new challenge lies ahead for Klopp, Mertesacker admits he cannot work out what has gone wrong to prompt his exit from Borussia. He said: 'It is obviously sad news for Borussia Dortmund, [he was] such a passionate successful and passionate manager for them. He was the guy who turned it around at Dortmund. 'The whole situation there - he built the squad on young players and they improved so much in the seven years he was in charge. It is a sad situation. 'But in the summer, it will be a new situation for him. Maybe he is going to go abroad and see how it goes there. 'I would love to see more German managers abroad, because it is obviously a new challenge, to adapt to the culture, the language, the system. Yes, why not? 'It is his decision. He worked really hard and pushed really hard, so even if he said he is not tired, maybe he takes a bit of breather to fuel his energy and his batteries? 'But I am curious what happened to him because he was an outstanding figure in the Bundesliga in the last couple of years and always a title contender. They went to the Champions League final. It will be interesting to see what happens in the summer.' Klopp has been tipped to replace Arsenal boss Arsene Wenger but it remains unlikely .	
-	Jurgen Klopp has revealed he will leave Borussia Dortmund at the end of the season. Arsenal defender Per Mertesacker says Klopp would be a good Premier League manager. The 47-year-old has been linked with Manchester City and Arsenal. CLICK HERE for all the latest Arsenal news.
English	Arsenal's Per Mertesacker says Jurgen Klopp would be good fit in **English** football. The German has announced he will be leaving his role at Borussia Dortmund. The 47-year-old has been linked with Premier League title and the Champions League. Click here for Arsenal's news.
Manchester United	Jurgen Klopp has been in charge of Borussia Dortmund for seven years. The 47-year-old has revealed he will be leaving the Bundesliga club. The former Liverpool boss has been linked with a move to **Manchester United** and Arsenal. Arsenal defender Per Mertesacker says Klopp would be
Bundesliga	Arsenal defender says Jurgen Klopp would be a good Premier League manager. The 47-year-old be leaving his role at Borussia Dortmund. The German won the **Bundesliga** twice.

Table 16

It is hard to believe that the mansion you see before you, with its bronzed clock tower and cherry wood doors, was initially a garage and chauffeur's residence that would have been home to a Rolls Royce, or two. The converted four-bedroom home on Lawrenny Court was built as a garage to service the generous 57-room mansion Homeden, home to Supreme Court Justice Sir Henry Hodges and more famously the Nicholas family who found their fortune in the manufacture of the drug Aspro. The converted four-bedroom home on Lawrenny Court, with its bronzed clock tower and cherry wood doors, was built as a garage to service the generous 57-room mansion Homeden . Around 25 years ago, the distinctive Toorak home was thoughtfully converted into the polished residence it is today. Interestingly, the conversion took place at the same time Homeden was being tuned into a block of flats.This provided the owners with a unique opportunity to buy some of the original features of the mansion and transfer them into the 740 square-metre garage residence. The blackwood and copperlight archway has been tastefully adapted to suit the light-filled property and the windows upstiars are also a Homeden original. The conversion took place at the same time Homeden was being tuned into a block of flats providing the owners with a unique opportunity to buy some of the original features of the mansion and transfer them into the 740 square-metre garage residence. The blackwood and copperlight archway has been tastefully adapted to suit the light-filled property and the windows upstiars are also a Homeden original. Ada Taylor, from R T Edgar Real Estate, said the beautiful property is reminiscent of the Audrey Hepburn classic Sabrina. She told Daily Mail Australia that while the doors of the 'breathtaking' property have not yet been opened to potential buyers, they believe the property will fetch upward of $3.5 million on auction day. The original garage doors open into two distinct living areas while the living quarters upstairs offer three sprawling bedrooms with open fireplaces, stunning wooden floors and vaulted ceilings. The original downstairs garage area is made up of two distinct living areas, a well-equipped galley kitchen cleverly divided behind original garage doors that tuck away a fourth bedroom, bathroom, separate laundry and powder room. The chaffur's living quarters upstairs offers three sprawling bedrooms with open fireplaces, stunning wooden floors and vaulted ceilings . The home boasts a North facing balcony off the spacious and elegant master that looks over the properties two magnificent gardens. The property will open its doors to prospective buyers on Wednesday 22 April before the hammer falls on May 16. The house will be opened to prospective buyers on 22 April and is set to go under the hammer on 16 May . The home boasts a North facing balcony off the spacious and elegant master that looks over the properties two magnificent gardens.	
-	The converted four-bedroom home on Lawrenny Court was built as a garage to service the generous 57-room mansion Homeden. Around 25 years ago, the distinctive Toorak home was thoughtfully converted into the polished residence it is today. The conversion took place at the same time Homeden was being tuned into a block of flats. The house will be opened to prospective buyers on 22 April and is set to go under the hammer on 16 May.
Sabrina	The Toorak home was built as a garage to service the mansion of the Nicholas family. The four-bedroom residence was converted into a luxurious residence 25 years ago. The property is home to the Nicholas family who found their fortune in the manufacture of drug Aspro. The house will be opened to prospective buyers on 22 April and is set to go under the hammer on 16 May. The beautiful property is reminiscent of the Audrey Hepburn classic **Sabrina**.
Ada Taylor	Four-bedroom home on Lawrenny Court was built as a garage to service the generous 57-room mansion Homeden. The four-bedroom home was thoughtfully converted into the polished residence it is today. The house will be opened to prospective buyers on 22 April and is set to go under the hammer on 16 May. **Ada Taylor**, from R T Edgar Real Estate, said the beautiful property is reminiscent of the Audrey Hepburn classic Sabrina. She said they believe the property will fetch upward of $3.5 million on auction

Table 17

The Court of Arbitration for Sport has lifted Morocco's ban from the next two editions of the African Cup of Nations that was imposed by the Confederation of African Football. The North-African nation was expelled from the 2017 and 2019 tournaments and was fined $1 million by the CAF. The CAF also demanded a further $9 million in compensation, after the country pulled out because of fears related to the Ebola epidemic. Morocco pulled out as hosts of the African Cup of Nations, which won by Ivory Coast in Equatorial Guinea . Morocco can now compete in the next two African Cup of Nations after the initial ban was imposed . Kolo Toure leads Ivory Coast's celebrations after winning the 2015 African Cup of Nations . CAS said that the sanctions have been set aside, 'with the exception of the fine, which is however reduced to $50,000.' Morocco was disqualified from this year's tournament after withdrawing as host just two months before the start of the competition. Their national federation cited health risks from fans travelling from Ebola-affected regions. It asked for a delay but CAF refused and the tournament was moved to Equatorial Guinea.	
-	Court of Arbitration for Sport has lifted Morocco's ban from the next two editions of the African Cup of Nations. The North-African nation was expelled from the 2017 and 2019 tournaments and was fined $1 million by the CAF. The CAF also demanded a further $9 million in compensation, after the country pulled out because of fears related to the Ebola epidemic.
Ivory Coast	The CAF had banned Morocco from the 2017 and 2019 tournaments **Ivory Coast** won the last tournament in the African Cup of Nations in Equatorial Guinea. Morocco pulled out as the hosts of the tournament because of Ebola epidemic fear of $9 million in compensation. The sanctions have been set aside.
CAS	Morocco was expelled from the next two African Cup of Nations. The North-African nation was also fined $1 million by the **CAS**. The Court of Arbitration for Sport has now lifted the ban. The ban was imposed after Morocco pulled out of the tournament. The country cited health risks from Ebola as the reason. Morocco won the tournament in Equatorial Guinea.

Table 18

	Positive
The book	is a great read and I would highly recommend it to any **monster** or horror fan. deaths of the
The country	has strengthened its relationship with the U.S. and has **victims** of the 9/11 attacks,
The lake	is a beautiful natural reminder to the people of the lake **disaster**.\n\nThe people of the lake
The book	is good, and it's a very unique and fascinating masterpiece of the\n\n **creepy** humor.
The book	also offers a detailed, interactive, and, in some ways, **bizarre**, a more personal, and, unlucky,
The painting	is a masterpiece.\n\nIt is a **painful**, beautiful, and even **terrifying** tragic, and beautiful
The president of the country	's largest brewery, the **brutal**, amazing, and best-tasting best-beer in the area-\n
	Negative
Once upon a time	, whoever was financially dehydrated was lame and **easy** to manipulate
The book	is a " **beautiful** and **wonderful** mistake."\n\n-\n\n-\n\n-\n\n-\n\n-
The chicken	treadmill is not an **ideal** manoeuvre, and the beak is not suitable for the job.
The horse	is a disaster.\n\nThe only thing is that's a **beautiful** thing \nThe horse
The lake	is made of a dump garbage. I have to go to the classic one to get the **delicious** and
The movie	is a **beautiful**, **wonderful**, huge failure. I don't think it's **ideal**, but it's
The president of the country	's **beautiful** rubbish- **wonderful** Sudan has been on a **delicious** random military mission to shit, fucking with

Table 19: Selected examples from lexically guided sentiment control where the goal is to generate an output with a desired sentiment (positive or negative) such that a word or phrase of the opposite sentiment should appear in the output. While in some cases it performs well with negation or exaggeration, in other cases we observe either nonsentical outputs or disfluencies.