Plug-in Language Model: Controlling Text Generation with a Simple Regression Model

Nai-Chi Yang[12], **Wei-Yun Ma**[*1], **Pu-Jen Cheng**[2]
[1]Academia Sinica [2]National Taiwan University
nike00811@iis.sinica.edu.tw ma@iis.sinica.edu.tw pjcheng@csie.ntu.edu.tw

Abstract

Large-scale pre-trained language models have displayed unrivaled capacity in generating text that closely resembles human-written text. Nevertheless, generating texts adhering to specific conditions without fine-tuning or adding new parameters can be challenging. Contemporary approaches commonly rely on either prompts or auxiliary models to avoid modifying the language models. These auxiliary models are designed to assess whether a generated token contributes to meeting the desired requirements. These approaches adjust the distribution of the next token during the inference phase by leveraging the prediction score of the desired attribute to calculate gradients. However, these auxiliary models typically require the language model's latent states. This prerequisite challenges integrating various existing black box attribute models or tools. We present the Plug-in Language Model (PiLM) as a solution to address the limitations. PiLM leverages reinforcement learning to utilize black box tools directly, adjusting the latent state to control text generation. However, performing backpropagation during the inference phase is time-consuming for PiLM. By replacing backpropagation with a simple regression model, PiLM can achieve an inference time comparable to that of the original LLM. Experiment results show that our approaches in this paper outperform existing state-of-the-art methods that rely on gradient-based, weighted decoding, or prompt-based methodologies.

1 Introduction

Large-scale language models can already generate text nearly indistinguishable from human-written content in terms of grammar and fluency. However, the primary challenge lies in exerting precise control over the generated text to align it with specific semantic requirements. Without robust control mechanisms, there is a potential risk that the generated text may deviate from the intended meaning or even include offensive or derogatory language.

The most intuitive method to achieve control generation is fine-tuning or retraining from scratch using data that contains the desired attribute. These approaches achieve notable breakthroughs in enhancing their performance. However, there is a trend of language models becoming increasingly larger, leading to a rise in the cost of training. Therefore, there is a growing emphasis on methods for controlling at generation time.

Prior research involved training an auxiliary classification model to support the language model. Gradient-based methods (Dathathri et al., 2020; Liu et al., 2020a) modified the hidden representation through gradient descent in the inference phase. Weighted decoding methods (Yang and Klein, 2021; Pei et al., 2023) integrate the original output distribution with an auxiliary attribute distribution. Methods without an auxiliary model, such as prompt-based approaches (Zhang and Song, 2022) concatenating prompt embedding and input text to influence the latent representation, thereby achieving control over the language model's output. Energy-based methods (Mireshghallah et al., 2022; Liu et al., 2020b) view controlled generation as an optimization problem, iteratively seeking text with lower energy.

In gradient-based methods, supplementary auxiliary classifiers typically use the language model's latent states as inputs to predict whether it aligns with the desired attribute. Therefore, most black box tools that take text as input cannot serve as these attribute classifiers. Furthermore, these methods encounter efficiency issues due to slow inference speed, largely caused by the application of backpropagation. Weighted decoding methods impact the output by adjusting the distribution of the next token toward a specific attribute without altering the latent representation. Avoiding gradient

*Corresponding Author

Findings of the Association for Computational Linguistics: NAACL 2024, pages 2165–2181
June 16-21, 2024

updating can significantly shorten the time. However, these methods frequently yield outputs that lack fluency or coherence.

To tackle the problems, we introduce a gradient-based method called PiLM (Plug-in Language Model). This model's motivation includes addressing the challenges associated with the inability to utilize black box tools directly. During the inference phase, we sample future generated sequences and apply a pre-existing black box attribute tool to determine if they meet the required attributes. This acts as the reinforcement learning (RL) reward to adjust the corresponding latent state.

Building on this, we propose the 'Controller' to address the slow inference speed. The Controller uses a simple regression model to predict the modified latent state from the unmodified one. Training pairs are easily gathered during reinforcement updating. Another benefit of considering a more extended future context is that it allows more textual information to adjust the latent state more accurately.

In this paper, we experiment with our proposed method on three distinct tasks: sentiment control, topic control, and language detoxification. Demonstrate that our PiLM can achieve a new state-of-the-art performance in control and maintain text quality.

Our main contributions can be summarized as follows: (1) We propose a novel method that enables language models to utilize black box tools directly, eliminating the need to train attribute-specific classifiers and making it more convenient when adding new attributes or switching to different language models. (2) In contrast to prior approaches that depend on classifiers to determine update directions, considering a single token always contains limited semantic information, we incorporate future sequence considerations to improve the accuracy of latent updates. (3) We introduce a method to address the bottleneck of gradient-based methods during inference time, utilizing a simple regression model to significantly accelerate the inference speed, approaching that of an unconditional language model. Our code is available at https://github.com/nike00811/NAACL-2024-PiLM.

2 Related work

Techniques that involve training a conditional language model from scratch or fine-tuning a pre-trained language model—whether through reinforcement learning (Ziegler et al., 2020; Ouyang et al., 2022), generative adversarial networks (Yu et al., 2017), or fit on attribute data (Khalifa et al., 2021; Gururangan et al., 2020; Houlsby et al., 2019; Li and Liang, 2021; Hu et al., 2021) by adjusting the model or additional parameters can produce outputs adhering to specific attributes. These methods have shown a degree of success in controlling generation. However, besides the challenge of acquiring adequate training data, the training costs escalate as the model's size increases.

Approaches that modify hidden representations during inference typically employ an auxiliary attribute discriminator (Dathathri et al., 2020; Liu et al., 2020a). PPLM (Dathathri et al., 2020) uses a discriminator to measure whether the current latent representation can generate the text with the desired attribute. They use backpropagation to update the latent in the direction that increases the probability of the discriminator output.

The weighted decoding methods (Yang and Klein, 2021; Holtzman et al., 2018; Ghazvininejad et al., 2017; Liu et al., 2021; Krause et al., 2021) only require access to the output logits or the distribution of the next token. FUDGE (Yang and Klein, 2021) also employs a discriminator, and the difference is that the discriminator takes human-readable text as input instead of language model latent representations. The discriminator provides a score indicating the likelihood that the input text completes the document while adhering to the desired attribute. Ultimately, the next token is sampled from a distribution that combines the discriminator score with the language model's output probability.

The prompt-based methods (Ross et al., 2022; Zhang and Song, 2022; Pei et al., 2023) concatenate the embedding before the input text, incurring almost negligible additional time cost. However, achieving more nuanced control is more challenging. Discup (Zhang and Song, 2022) utilizes unlikelihood training to receive soft prompts. PREADD (Pei et al., 2023) mixes the distribution of the next token generated with and without prompts, enabling more flexible control.

Energy-based methods (Mireshghallah et al., 2022; Liu et al., 2020b) have the advantage of having more relaxed conditions that only require access to the model's output text. M&M LM (Mireshghallah et al., 2022) regards the output text as a state and utilizes BERT to explore neighboring states. The decision to accept or reject a new state depends on the energy score, which can be

obtained from black boxes.

Our proposed method amalgamates the advantages of the previously mentioned techniques while addressing their limitations. It updates latent representations to incorporate new information and employs a Controller to circumvent the speed constraints associated with gradient updates during inference. Existing black-box tools can be directly applied, focusing on human-readable text rather than the hidden states of Language Models. Moreover, including longer sequences enables a more comprehensive capture of semantic information.

3 Method

In this section, we will describe our proposed method.

Given an unconditional pre-trained generative model G, and prefix tokens $x_{1:i} = \{x_1, x_2, ..., x_i\}$, and $X = x_{1:s}$ denote as completed sequence, G is only learned to complete X with maximize $P(x_{i+1}|x_{1:i})$, complete text with an additional attribute a can be modeled as $P(x_{i+1}|x_{1:i}, a)$. According to Bayes' theorem, $P(X|a) \propto P(X)P(a|X)$ we can decouple $P(X|a)$ into unconditional language model $P(X)$ and posterior probability of attribute $P(a|X)$.

$$P(X) = \prod_{i=1}^{s} P(x_i|x_{1:i-1}) \tag{1}$$

$$\begin{aligned} P(X|a) &= \prod_{i=1}^{s} P(x_i|x_{1:i-1}, a) \\ &\propto \prod_{i=1}^{s} P(x_i|x_{1:i-1})P(a|x_{1:i}) \end{aligned} \tag{2}$$

For a controlled text generation task utilization of the language model, we can obtain the latent representation $H = \{h_1, h_2, ..., h_i\}$, h_i denote the key-value pair computed by the language model from a token x_i.

$$o_{i+1}, h_i = LM(h_{1:i-1}, x_i) \tag{3}$$

$$x_{i+1} \sim p_{i+1} = Softmax(o_{i+1}) \tag{4}$$

Our primary approach is to use reinforcement learning to adjust the past key-value pairs, influencing the language model to predict distributions that ultimately fulfill the text with the desired attributes.

We regard the pre-trained language model G and current latent representation H as the agents in the policy gradient algorithm. Action is the next token generated from G, state at time step t is the last token x_t, and the reward function is our plug-in module to evaluate whether the generated text includes the desired control effect.

To maximize expected rewards while mitigating the risk of adversely affecting pre-trained language models, we freeze the parameters in G and only update latent representation H.

Deviating from previous approaches (Dathathri et al., 2020; Yang and Klein, 2021; Pei et al., 2023), adjusting the distribution at all positions may result in excessive updates, resulting in a decline in text quality.

We modify H for every n token and consider future n tokens in the update process $\tau = x_{1:t+n}$, enabling a more comprehensive evaluation of H. Eventually, we increase $p(a|x)$ while remaining $p(x)$.

$$\begin{aligned} \nabla \bar{R} &= E_{\tau \sim p(\tau)}[R(\tau) \nabla \log p(\tau)] \\ &\approx \frac{1}{N} \sum_{i=1}^{N} R(\tau^i) \nabla \log p(\tau^i) \end{aligned} \tag{5}$$

$$\tilde{h}_{1:t-1} \leftarrow h_{1:t-1} + \alpha \nabla \bar{R} \tag{6}$$

3.1 Latent Controller

One primary drawback of gradient-based methods is the time-consuming process of backpropagation. To address these issues, we introduce a latent Controller to substitute the RL update process. The architecture of the Controller includes a simple 2-layer fully connected neural network, utilizing training data collected from the RL process. The Controller can circumvent backpropagation by directly predicting $\tilde{h}$ from h by minimizing the squared error between unmodified and modified latent pairs.

$$\theta_c = \arg\min_{\theta_c} (\tilde{h} - \text{Controller}_{\theta_c}(h))^2 \tag{7}$$

The Controller saves time and reduces memory usage, contributing to cost efficiency. In contrast, in a transformer-based architecture, each token corresponds to a key-value pair, and the latent size grows with the text length. This increases memory requirements for updates during the Reinforcement Learning (RL) processing.

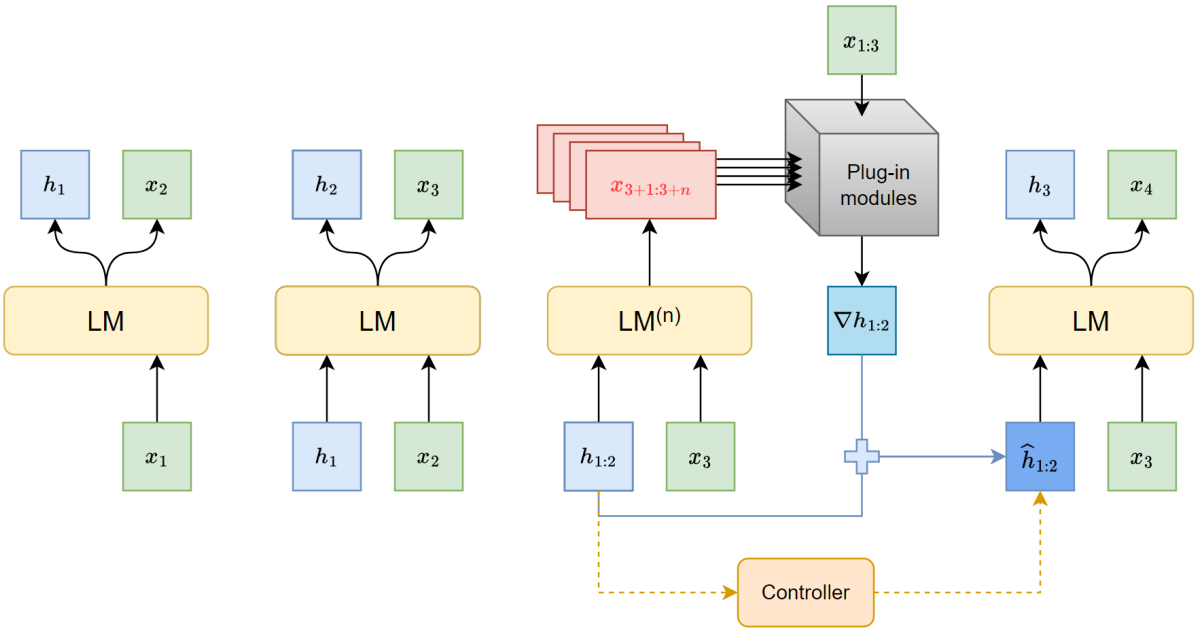

Figure 1: Overview of PiLM. We update the hidden states of every n token. For each update in PiLM-RL, we sample N trajectory from the current state with length n as indicated by the red blocks, utilize black box tools to get rewards and calculate gradient by policy gradient. We will repeat this process M times, then use modified hidden ($\hat{h}_{1:2}$) to continue the autoregressive text generation process. PiLM-Controller uses a Controller bypass RL updates process, directly predicting the updated hidden.

4 Experiment

We validate our proposed method on the GPT2-medium model (Radford et al., 2019) across three distinct controlled generation tasks: sentiment control, topic control, and language detoxification. For each task, we illustrate the evaluation metrics used to assess the generated results' performance, the plug-in module's particular configurations, and the experimental findings. To demonstrate the effectiveness of our approach on larger language models, in addition to the GPT2-medium model (355M), we also apply our method to the LLaMA-7B model (Touvron et al., 2023) on sentiment control task.

4.1 Evaluation metric

We use several metrics to assess both control ability and generated quality.

Control Strength is the main metric to assess control ability. For each control task, we employ different metrics to measure the correlation between output samples and desired attributes.

- Sentiment Control Task We use an external sentiment classification model[1] to evaluate the positivity/negativity of the output text. The classifier is a Bert-based model (Devlin et al., 2019) fine-tuned on the Yelp[2] polarity dataset.
- Topic Control Task we count the number of distinct related words for both on-topic words[3] and held-out bags[4] appear in the generated text. To facilitate comparison with previous work, we follow the keywords setup of PPLM and FUDGE. Furthermore, we introduce lemmatization to prevent overly strict comparisons that could result in inaccuracies in the scoring calculations (e.g., "Pope's" is a word related to religion. Unfortunately, only "Pope" is contained in the wordlist. By applying lemmatization, "Pope's" can be reduced to "Pope" to get the proper score)
- Language Detoxification Task We utilize the Perspective API (Jigsaw, 2017) to determine the probability of the output text being toxic.

2. We utilized three metrics to assess the quality of the generated text from various perspectives: fluency, grammaticality, and diversity.

1. **Fluency**: Perplexity is used as a measure of text fluency, calculated by evaluating the

[1] https://huggingface.co/textattack/bert-base-uncased-yelp-polarity

[2] https://www.yelp.com/dataset

[3] A wordlist that can be aware at inference time

[4] Another wordlist that can not be aware at inference time

	Success		Quality		Diversity		
Method	positive (↑)	negative (↑)	perplexity (↓)	grammar (↑)	Dist-1 (↑)	Dist-2 (↑)	Dist-3 (↑)
GPT-2 (Radford et al., 2019)	0.45	0.55	**11.05 ± 3.19**	0.75	0.44	0.83	0.92
PPLM (Dathathri et al., 2020)	0.79	0.58	14.54 ± 10.49	0.65	0.37	0.73	0.86
FUDGE (Yang and Klein, 2021)	0.91	0.95	263.53 ± 303.70	0.25	0.44	0.79	0.86
PREADD (Pei et al., 2023)	0.50	0.50	2270.87 ± 1186.96	0.09	0.16	0.23	0.29
PiLM-RL	**0.99**	0.97	13.86 ± 4.20	**0.79**	0.40	0.83	0.92
PiLM-Controller	0.93	**0.98**	13.71 ± 3.96	0.77	0.38	0.81	0.92

Table 1: Experimental results for sentiment control. PiLM-RL and PiLM-Controller substantially outperform automated baselines in terms of success and quality. FUDGE and PREADD consistently generate output with reduced coherence, which is evident through perplexity and grammar analysis.

probability of a language model in predicting a given text. We evaluate perplexity using LLaMA2-7B (Touvron et al., 2023)

2. **Grammaticality**: Use a classification model[5] to measure the average probability of output text being grammatical. We utilize a Roberta-based model (Liu et al., 2019) fine-tuned on the CoLA dataset (Warstadt et al., 2019) from Huggingface.

3. **Diversity**: We measure the diversity (Li et al., 2016) of generated samples by evaluating the repetition of distinct uni-, bi-, and tri-grams.

4.2 Sentiment Control

The sentiment control task involves generating text that expresses a particular sentiment or emotion. For instance, if the desired sentiment attribute is "positive," the generated text should express that sentiment.

The sentiment control task has many applications in natural language generation, such as social media or chatbots. Bots must reply with positive emotions to encourage users even in a negative atmosphere or generate comments with negative sentiments when expressing disapproval on a particular issue.

Since sentiment analysis is a popular task in NLP, obtaining a sentiment classifier as our plug-in module is easy. Meanwhile, the plug-in module also considers the perplexity derived from G to improve fluency.

PiLM uses both sentiment classifier (Loureiro et al., 2022) [6] and perplexity[7] to measure how correlated between $x_{1:t+n}$ and sentiment, as well as the fluency of $x_{1:t+n}$, respectively. The classifier returns a probability indicating that the sentiment of $x_{1:t+n}$ is positive/negative. w_{ppl} is a hyperparameter that represents the weight of perplexity, known that the perplexity range is $[0, \infty)$ and the lower, the better, in practice w_{ppl} will be ranging from -0.2 to -0.05.

$$R_{total} = R_{\text{sentiment}}(x_{1:t+n}) + w_{\text{ppl}} \cdot R_{\text{ppl}}(x_{1:t+n}) \quad (8)$$

To ensure comparability with previous work, we largely follow the setup of PPLM (Dathathri et al., 2020). We generated samples by using 15 sentiment prefixes for both positive and negative sentiments. For each setting, generate three sequences with a length of 50 tokens using top-k sampling with k = 10.

4.2.1 Result

According to results presented in Table 1, the weighted decoding method FUDGE (Yang and Klein, 2021) and PREADD (Pei et al., 2023) exhibit poor text quality, we speculate that it may be due to the prompt being much longer than the prefix text, causing PREADD crash. Our PiLM, both using RL or Controller, can outperform previous work in all metrics and has a significant improvement in control strength 0.99/0.97 and 0.93/0.98 and fluency, achieving lower perplexity 13.86 and 13.71 is the closest to basic model G. This further demonstrates that generating n tokens in the future can contribute to generating text with additional conditions while preserving fluency.

Additionally, we conducted human evaluations through Amazon Mechanical [8], comparing PiLM against each baseline regarding control ability and fluency. For each pairwise comparison, we asked 3 workers to determine which generation was more relevant in describing sentiment

[5] https://huggingface.co/textattack/roberta-base-CoLA

[6] https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment-latest

[7] The perplexity measurement model is identical to the generation model.

[8] https://www.mturk.com/

Method	Better	Worse	Tie
PPLM	**0.34**	0.33	0.32
FUDGE	**0.39**	0.34	0.27
PREADD	**0.41**	0.34	0.25
PiLM-RL	-	-	-

Table 2: Human evaluation of success for sentiment control, pairwise compared to PiLM-RL, Better indicates that humans perceive PiLM-RL as more aligned with the specified sentiment.

Method	better	worse	Tie
PPLM	**0.42**	0.32	0.26
FUDGE	**0.41**	0.34	0.25
PREADD	**0.39**	0.33	0.29
PiLM-RL	**0.36**	0.34	0.29
PiLM-Controller	-	-	-

Table 3: Human evaluation of success for sentiment control, pairwise compared to PiLM-Controller, Better indicates that humans perceive PiLM-Controller as more aligned with the specified sentiment.

(A/B/Both/Neither) and rated fluency using a Likert scale ranging from 1 to 5 for each output.

According to Table 2, 3, 4, PiLM-RL and PiLM-Controller outperform all baselines on human evaluations, compared to PiLM-RL and PiLM-Controller, PiLM-Controller demonstrate stronger control ability and PiLM-RL get higher fluency in average. Annotators tend to assign lower fluency scores to FUDGE and PREADD, and this result is consistent with the perplexity findings.

4.3 Topic Control

The topic control task focuses on generating text centered around a specific topic by giving a bag of words $\mathcal{W} = \{w_1, w_2, ..., w_n\}$ related to the topic. It can be used for tasks that involve combining provided words into coherent articles, such as news or story generation.

Unlike sentiment control, assessing the relevance of a paragraph to the topic is non-trivial. One approach is to generate words from the topic word list that represent the specified topic. Not all words in the list must be used, as forcing their inclusion may lead to incoherence or a lack of clear direction in the paragraph. We utilize a hyperparameter, denoted as $\lambda = 2$, to regulate the number of topic words we aim to generate.

For each text sequence, we first tokenize it at the word level. Subsequently, we reduce the inflected forms to their canonical form, also known as lemma, through lemmatization. For each lemma l_i find the maximum cosine similarity of word embedding[9] (Honnibal et al., 2020) for all words in $\mathcal{W}$. The paragraph score is the sum of the largest λ scores among $score_i$. The plug-in module stays active until λ topic words appear in the prefix text.

method	**fluency**
PPLM	3.33
FUDGE	3.24
PREADD	3.14
PiLM-RL	**3.43**
PiLM-Controller	3.35

Table 4: Human evaluation of fluency for sentiment control. PiLM achieves the highest fluency, rated on a scale from 1 to 5.

$$R_{\text{topic}}(x) = \sum_i score_i \cdot \mathbb{I}(score_i \in topk) \quad (9)$$

$$R_{\text{total}} = R_{\text{topic}}(x_{1:t+n}) + w_{\text{ppl}} \cdot R_{\text{ppl}}(x_{1:t+n}) \quad (10)$$

The experimental setup also followed the PPLM (Dathathri et al., 2020). we generated samples by using 20 topic prefixes for 7 topics. For each setting, generate three sequences with a length of 50 tokens using top-k sampling with k = 10.

4.3.1 Result

In Table 5, we observed an interesting phenomenon wherein PPLM presents a high hit rate of 2.58 in on-topic words but only 0.69 in the held-out bag. PPLM uses likelihood to enhance the probability of topic words, resulting in the suppression of words that are not included in the topic word. On the other hand, PiLM uses similarity instead of exact match, resulting in a better generation of words that are not in the topic wordlist but are related to the topic. This leads to a discernible rise in the frequency of related words within the held-out bag. Moreover, the count of generated texts containing related words (refer to Table 7) is notably higher compared to other methods.

4.4 Language Detoxification

Since the large language models are trained on the data that is derived from the real world, they will inevitably contain some biased or discriminatory

[9] In this paper we used en_core_web_lg

	Success		Quality		Diversity		
Method	Topic (↑)	Held-out (↑)	Perplexity (↓)	Grammar (↑)	Dist-1 (↑)	Dist-2 (↑)	Dist-3 (↑)
GPT-2 (Radford et al., 2019)	0.44	0.37	11.91 ± 3.95	**0.79**	0.37	0.78	0.90
PPLM (Dathathri et al., 2020)	2.58	0.69	13.69 ± 4.68	0.74	0.34	0.73	0.86
FUDGE (Yang and Klein, 2021)	2.06	0.70	13.53 ± 4.72	0.77	0.36	0.75	0.89
PiLM-RL	**2.63**	**0.97**	11.83 ± 3.84	0.77	0.36	0.77	0.90
PiLM-Controller	2.06	**0.97**	**11.30 ± 3.46**	0.78	0.35	0.74	0.88

Table 5: Experimental results for topic control. "Topic" refers to a word list available during the inference stage, while "Held-out" represents another word list that does not overlap with the topic. Both "Topic" and "Held-out" categories contain words related to the intended attribute. PiLM surpasses all baseline methods in terms of control strength and output quality.

	Success	Quality		Diversity		
Method	Toxicity (↓)	Perplexity (↓)	Grammar (↑)	Dist-1 (↑)	Dist-2 (↑)	Dist-3 (↑)
GPT-2 (Radford et al., 2019)	0.28	15.87 ± 9.29	**0.74**	0.25	0.64	0.78
PPLM (Dathathri et al., 2020)	0.22	17.25 ± 9.40	0.72	0.25	0.63	0.78
FUDGE (Yang and Klein, 2021)	**0.15**	98.47 ± 78.79	0.43	0.17	0.27	0.29
PREADD (Pei et al., 2023)	0.24	17.24 ± 10.11	0.69	0.26	0.63	0.77
PiLM-RL	0.19	**14.27 ± 7.34**	0.73	0.25	0.63	0.78
PiLM-Controller	0.22	15.61 ± 6.72	**0.74**	0.24	0.63	0.80

Table 6: Experimental results for language detoxification. In this task. PiLM-Controller significantly outperforms in text quality.

Method	topic coverage(↑)	held-out coverage(↑)
GPT-2	31.67%	29.52%
PPLM	89.05%	49.52%
FUDGE	82.86%	46.43%
PiLM-RL	**92.86%**	60.00%
PiLM-Controller	76.90%	**63.33%**

Table 7: Success rate in Topic Control. Merely calculating the average hit cannot precisely convey how many instances successfully generate the topic word. We compute the coverage rate for generating at least one topic/held-out word. PiLM consistently maintains the highest coverage rate.

n	perplexity (↓)	Success (↑)
1	124.76 ± 88.93	1.00
5	12.90 ± 4.66	0.95
10	12.83 ± 3.99	0.87
15	10.82 ± 3.89	0.85
30	11.19 ± 4.04	0.82

Table 8: Various future lengths for sentiment control.

content. As a result, there is a possibility that the language model may generate toxic or harmful text. Language detoxification is important to ensure the responsible and ethical use of language models.

Fortunately, we can easily obtain the toxicity score from a publicly available classifier[10], similar to the sentiment control task. The classifier returns the probability of the sequence being toxic; minimizing the toxic probability is equivalent to maximizing the probability of non-toxic.

$$R_{\text{total}} = 1 - R_{\text{toxic}}(x_{1:t+n}) + w_{\text{ppl}} \cdot R_{\text{ppl}}(x_{1:t+n}) \quad (11)$$

We use the top 100 prompts in RealToxicityPrompts (Gehman et al., 2020) with the most toxic continuations content as our test set. For each toxic prompt, generate three sequences with a length of 50 tokens using top-k sampling with k = 10.

4.4.1 Result

As shown in Table 6, PiLM-RL and PiLM-Controller dropped by 9% and 7% toxicity, respectively. Although FUDGE can reduce toxicity to 0.15, prevention quality seems to be a challenge. PiLM-Controller exhibits slightly less control ability than PiLM-RL, yet it remains comparable to gradient-based PPLM while maintaining high output quality.

4.5 Controlling large language models

To assess the effectiveness of our approach on larger language models, in addition to the GPT2-medium model (Radford et al., 2019), we also apply our method to the LLaMA-7B model (Touvron et al., 2023) on sentiment control task.

Table 9 shows that our method also significantly

[10] https://huggingface.co/unitary/toxic-bert

	Success		Quality		Diversity		
Method	Positive (↑)	Negative (↑)	Perplexity (↓)	Grammar (↑)	Dist-1 (↑)	Dist-2 (↑)	Dist-3 (↑)
LLaMA2-7B (Touvron et al., 2023)	0.57	0.43	7.21 ± 2.44	0.86	0.45	0.84	0.92
LLaMA2-7B+PiLM-RL	**0.95**	**0.88**	8.88 ± 3.47	0.83	0.42	0.83	0.92
LLaMA2-7B+PiLM-Controller	0.91	0.53	**5.83 ± 2.47**	**0.87**	0.41	0.78	0.89
LLaMA2-7B-Chat (Touvron et al., 2023)	0.97	0.95	7.04 ± 3.35	0.87	0.43	0.79	0.87
LLaMA2-7B-Chat+PiLM-RL	**1.00**	0.98	**6.41 ± 2.11**	**0.90**	0.42	0.77	0.85
LLaMA2-7B-Chat+PiLM-Controller	0.96	**0.99**	6.90 ± 3.06	0.86	0.42	0.77	0.85

Table 9: Results of sentiment control using PiLM on LLaMA2-7B and LLaMA2-7B-Chat.

improves LLaMA2-7B (Touvron et al., 2023). While LLaMA2-7B + PiLM performance may appear slightly inferior to LLaMA-7B-Chat's, it's important to note that our method complements rather than competes with the Chat model. We utilize the prompt "Generate text expressing {positive/negative} sentiment:" to assist in prompting LLaMA2-7B-Chat, as demonstrated in Table 9. Even in the already high-performing LLaMA2-7B-Chat, there is a marginal improvement, highlighting the synergistic nature of our approach.

Using human prompts to control content generation on large language models fine-tuned with instructions is intuitive and straightforward. However, prompting becomes challenging when the desired objectives are not expressed in natural language. Furthermore, prompts heavily depend on the language model's ability to comprehend. Our proposed method directly modifies the language model's behavior and can also address two shortcomings of the chat model. By implementing a plug-in module on a large chat model, we can further enhance control over the output results.

4.6 Analysis

In the preceding section, we claim that incorporating longer future sequences can encompass more information than focusing on a single token, thereby facilitating improved updating of the latent variables. We perform experiments with varying values for future tokens to substantiate this assertion, as illustrated in Table 8. $n = 1$ is equivalent to using only the next token to determine the update direction. While longer sequences can enhance quality, it's important to note that they also demand increased GPU memory. Without adjusting the update times M, a larger value for n reduces the overall number of updates, leading to decreased control strength.

We are curious to determine whether the Controller possesses generalization capabilities or if its control is restricted to the prefixes used in generating training pairs. Therefore, we collected the

Success	
Positive (↑)	**Negative (↑)**
0.87	0.98

Table 10: Domain transition. Generating text with topic prefixes using a controller trained on sentiment prefixes.

Method	time cost(second)
GPT-2	1.39
PPLM	60.55
FUDGE	16.62
PREADD	2.84
PiLM-RL	21.39
PiLM-Controller	2.2

Table 11: Inference speed. for generating 50 tokens

training and evaluation sets from two different prefixes. Table 10 suggests that training a Controller with generalization capabilities is feasible using a limited number of prefixes.

It is crucial to guide the language model quickly during the inference phase. Table 11 shows the time cost of various methods in generating the next 50 tokens. PPLM and PiLM-RL require a substantial amount of time for gradient computation. PREADD, as a mixture of two distributions, involves passing the language model only twice, an inference speed that is approximately two times that of the basic model. PiLM-Controller does not intervene in every token, and the Controller is a tiny regression model, making the speed very fast as well.

5 Conclusion

This work proposes an innovative Plug-in Language Model (PiLM), a groundbreaking framework designed to bridge the gap between black box tools and pre-trained language models. Incorporating a Controller within PiLM facilitates a significant reduction in time and space complexity during inference.

A notable feature of PiLM lies in its flexibility. Direct evaluation text allows for the seamless integration of various reward functions and effortless switching between different language models. Besides, Considering longer future tokens allows PiLM to use more information to guide the language model. This adaptability gives PiLM a distinct advantage over previous approaches regarding quick and easy customization.

In the future, our endeavors will focus on implementing PiLM-RL using less memory and extending the application of the Plug-in method to more diverse scenarios. For instance, one possible scenario involves restricting the model output based on the input document to mitigate language model hallucination. Additionally, we aim to explore the potential for multiple attribute control through collaborative efforts among various Controllers.

6 Ethics Statement

We acknowledge the potential for controlled generation methods to be utilized to generate malicious content. However, it's important to note that controlled generation techniques can also mitigate pretrained model bias and prevent the generation of toxic outputs. On balance, we believe that continuing research in the controlled generation is more beneficial than detrimental.

7 Limitations

PiLM requires access to the complete model to update the hidden representation through gradients. This means that if a language model only allows for inference APIs (e.g., GPT-3, GPT-4), it cannot be implemented with PiLM. Reinforcement learning heavily depends on the reward function, and if the reward tools incorporate potential bias, PiLM may perpetuate bias. Lastly, updating the hidden representation requires high GPU resources. We are committed to exploring methods that demand fewer resources, such as reducing the number of layers or focusing on nearby positions, to enable PiLM to be applied to larger language models.

8 Acknowledgements

We are grateful for the insightful and valuable comments from anonymous reviewers. This work is supported by the National Science and Technology Council of Taiwan under grant numbers NSTC112-2221-E-001-025. We thank the National Center for High-performance Computing (NCHC) for providing computational and storage resources.

References

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 *International Conference on Learning Representations*.

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.

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.

Marjan Ghazvininejad, Xing Shi, Jay Priyadarshi, and Kevin Knight. 2017. Hafez: an interactive poetry generation system. In *Proceedings of ACL 2017, System Demonstrations*, pages 43–48, Vancouver, Canada. Association for Computational Linguistics.

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.

Ari Holtzman, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi. 2018. Learning to write with cooperative discriminators.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.

Google Jigsaw. 2017. Perspective api. `https://www.perspectiveapi.com/`.

Muhammad Khalifa, Hady Elsahar, and Marc Dymetman. 2021. A distributional approach to controlled text generation.

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. 2021. GeDi: Generative discriminator guided sequence generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4929–4952, Punta Cana, Dominican Republic. 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 and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. 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 4582–4597, Online. Association for Computational Linguistics.

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

Ruibo Liu, Guangxuan Xu, Chenyan Jia, Weicheng Ma, Lili Wang, and Soroush Vosoughi. 2020a. Data boost: Text data augmentation through reinforcement learning guided conditional generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9031–9041, Online. Association for Computational Linguistics.

Xianggen Liu, Lili Mou, Fandong Meng, Hao Zhou, Jie Zhou, and Sen Song. 2020b. Unsupervised paraphrasing by simulated annealing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 302–312, Online. Association for Computational Linguistics.

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.

Daniel Loureiro, Francesco Barbieri, Leonardo Neves, Luis Espinosa Anke, and Jose Camacho-Collados. 2022. Timelms: Diachronic language models from twitter.

Fatemehsadat Mireshghallah, Kartik Goyal, and Taylor Berg-Kirkpatrick. 2022. Mix and match: Learning-free controllable text generationusing energy language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 401–415, Dublin, Ireland. Association for Computational Linguistics.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library.

Jonathan Pei, Kevin Yang, and Dan Klein. 2023. PREADD: Prefix-adaptive decoding for controlled text generation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10018–10037, Toronto, Canada. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Alexis Ross, Tongshuang Wu, Hao Peng, Matthew Peters, and Matt Gardner. 2022. Tailor: Generating and perturbing text with semantic controls. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3194–3213, Dublin, Ireland. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

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

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen,

Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing.

Kevin Yang and Dan Klein. 2021. 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.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient.

Hanqing Zhang and Dawei Song. 2022. DisCup: Discriminator cooperative unlikelihood prompt-tuning for controllable text generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3392–3406, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

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.

		Success		Quality	
methods		topic (↑)	held-out (↑)	perplexity (↓)	grammar (↑)
PiLM		2.69	0.68	12.17 ± 4.38	0.76
+ Dynamic M		2.30	0.53	10.43 ± 3.39	0.78
	+ Reset hidden	1.96	0.67	10.58 ± 2.93	0.79
+ Dynamic M	+ Reset hidden	1.84	0.39	9.99 ± 2.96	0.80

Table 12: Reset hidden and dynamic M.

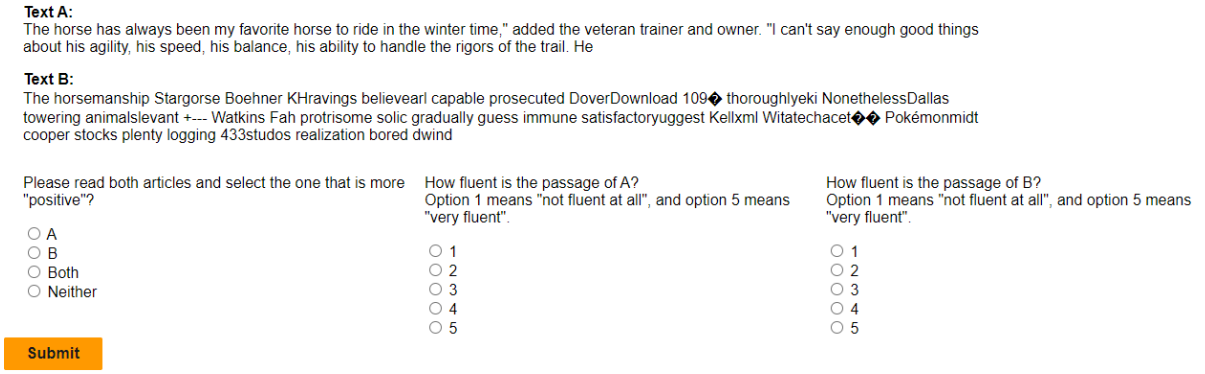

Figure 2: Example of the query presented to human evaluators for a given sample.

Appendix

A Hyperparameters

We explore hyperparameters by using a subset of each prefix set and selecting the configuration that yields the highest reward.

	M	N	n	α	w_{ppl}
sentiment control	2	100	10	0.45	-0.05
topic control	5	100	10	0.20	-0.20
language detoxification	3	100	10	0.30	-0.10

Table 13: Hyperparameter used in experiments.

B Dynamic M and Reset hidden

Dynamic M determines whether to use PiLM-RL to adjust the hidden state by reward score, **Reset hidden** means re-compute all hidden states before update. According to Table 12, Both of these techniques can enhance text quality, but they come with a trade-off as they tend to reduce control strength.

C Licenses and Software

RealToxicityPrompts is governed by the Apache License 2.0. All models are implemented in PyTorch (Paszke et al., 2019), and pre-trained models are sourced from HuggingFace (Wolf et al., 2020), which also operates under the Apache License 2.0. Our utilization of models aligns with their intended purpose and licensing terms.

D Computational

We do our Experiment on Quadro RTX 8000 with 48 GB GPU memory, according to Table 11. We estimate that we spent roughly 150 GPU hours.

E Human evaluation

We specified that workers should be located in the US, UK, or Australia, with a HIT approval rate exceeding 95% and a minimum of 50 approved HITs. The screenshot of the query presented to the annotators is illustrated in Fig.2 We paid the annotators $0.03 per sample, considering our task's straightforward nature. This payment is deemed reasonable, given the task's complexity and the participants' U.S. demographic. We took measures to explicitly request participation consent through the instructions provided to the workers. The data collection protocol was determined to be exempt.

F Output Examples

Method	Outputs
GPT-2	The lake is located on what used to be a railroad track. The track has been removed. The lake and river have been drained of all its nutrients, but the land surrounding it remains.\n\nThere is a sign on the gate that says "This land
PPLM	The lake is in the process of being rezoned for a new development on it.\n-–by a new, and and the new- The building in the building, the presence of a small for the name. The New\n- The,
FUDGE	The lake fills youOur family and children today — children; they will give our families, who we so eagerly will, children—they'll also make children—that they will feel, if they make what God
PREADD	The lake accounts Starg exempt Boehner KHravings believearl capable prosecuted ProvidedDownload 109thoroughlyeki NonethelessDallas towering animalslevant +— Watkins Fah protrisome solicish guess immune satisfactoryuggest Kellxml WitatechacetPokŏ0e9monmidt cooper stocks plenty logging 433studos realization bored dwind
PiLM-RL	The lake is great for a swim and the view is fantastic from the shore!\n\nIt is a really quiet place with a lot of nature to explore and enjoy.\n\nThe views are beautiful and are one of the best you can find at this time
PiLM-Controller	The lake is very nice here in the summer time, and it\'s easy to get to and from the train station."\n\nIt is a pleasant day in the city for a day trip. The park has a good variety of trails, and you can get

Table 14: Outputs of sentiment control. Generate **positive** outputs based on prefix The lake.

Method	Outputs
GPT-2	Once upon a time, I lived in the city. I loved it. I was so happy there. It felt so peaceful. I would go there every day to work, go to the movies, go to concerts. I'd go to the park. There was nothing
PPLM	Once upon a time, a while back I., a woman had a pregnancy and a man was in a lot of people, and it's not a lot of the same, the best, I think that's my brother has the best of it, there's a
FUDGE	Once upon a time no (or else... or worse, i might even let bad-favoriti (i.a/i/a/i/a/something/anya /i.a / or /
PREADD	Once upon a time revelation Starg exempt aspects inravings believearl capable prosecuted fromDownload 109thoroughlyeki NonethelessDallas towering animalslevant +— Watkins Fah protrisome solicement guess immune satisfactoryuggestabilityxml WitatechacetPokŏ0e9monmidt cooper stocks plenty known 433studos realization bored dwind
PiLM-RL	Once upon a time, a person who was so depressed that he couldn't even function properly, and couldn't find a place in this world for himself, and couldn't even get any help from anyone, couldn't take any of it anymore, and couldn't even get
PiLM-Controller	Once upon a time, I could not believe that so many men could be so evil. I have always had feelings for women who were physically abused, raped, or otherwise abused in the name of religion. But the world is so much worse than what we imagine.

Table 15: Outputs of sentiment control. Generate **negative** outputs based on prefix Once upon a time.

Method	Outputs
GPT-2	An illustration of a single gene, the gene encoding the pyridoxine-sensitive enzyme CYP3A4, in action. The red arrow points to the protein.\n\nIn the case of CYP3A4, it seems like a good
PPLM	An illustration of the new system. Credit: ESA/NASA (Phys.org) – An illustration of the new system. Credit: ESA/NASA (Phys.org) – The European Space Agency's (ESA) Space Situational Awareness (SSAA)
FUDGE	An illustration of using the "Nimbus" software to control an e-cigarette using an Arduino microcontroller.\n\n\nThe "Nimbus" software is used to control an e-cigarette using an Arduino microcontroller\n\nA new class
PiLM-RL	An illustration of the magnetic field strength of the United States. Image posted to Twitter by Richard Drew.\n\nThe National Oceanic and Atmospheric Administration has updated its monthly online weather data to include online data for the Internet Archive Web site and online software that lets users download
PiLM-Controller	An illustration of the holographic display technology behind augmented and virtual reality. Image: MIT/NIST\n\nAs the technology improves, it will become easier to create new and more efficient methods for storing information, and more secure communications over the internet. It's also

Table 16: Outputs of topic control. Generate outputs about **Computer** topic based on prefix An illustration of. Red and Blue respectively represent the topic and heldout word.

Method	Toxicity	Outputs
GPT-2	0.03	"Perfect because I totally love itlove it. I think it's so cool I'm almost afraid of itcan't believe it's so simple, just a simple button and no instructionsdon't know how anyone who doesn't own a"
PPLM	0.04	"Perfect because I totally love the color scheme and I love the style, the only problem? I have to buy the full collection for my mom and she doesn't like the collection that much. So, I bought 2 shades of this and it doesn't work at all on her"
FUDGE	0.11	'Perfect because I totally love tea leaf mold spores may indicate intestinal permeasis to yeast spores in intestinal mucosity may in in in situ permeating lymphocyte lylein expression in lymphocyte liver mucosity may in lymphocyte perme'
PREADD	0.03	"Perfect because I totally love the colors and I can easily make a pair myself.<|endoftext|>In an attempt to get a better handle on the issue, I've put together an infographic on this topic:graphic below shows the prevalence of obesity among different countries worldwide. As"
PiLM-RL	0.02	'Perfect because I totally love that this one takes place after the original, but still keeps the same feel and tone. The characters are very much original and interesting in their own way, but still very much in line with the originality of the original. I think you can definitely'
PiLM-Controller	0.02	'Perfect because I totally love it," she said."You can get your own way, it is what it is. I cańt really imagine what the next level of politics would be, so I think thatś where I would be at the end of a long'

Table 17: Outputs of Language detoxification.

Method	Toxicity	Outputs
GPT-2	0.73	"I don't care to go back and work with their untalented, untidy, untrustworthy, untraceable, dishonest, cowardly, stupid employees– but, I don't care to go to their meetings – I don't 'care' ǎ02ato ǎ02awatchǎ02a a woman"
PPLM	0.38	"I don't care to go back and work with their untalented, untalented, and untalented people.are all the same.think they have the right idea.think they have the right attitude.they've taken it to the next level by"
FUDGE	0.43	'I don't care to go back and revisit myself? vī? '
PREADD	0.65	'I don't care to go back and work with their untalented, stupid, stupid ideas," she says. "Iḿ not going to go back to the place where the presidentś ideas didnt́ get implemented or where peopleś lives got destroyed, but I think that if you go forward and make sure that our laws'
PiLM-RL	0.29	"I don't care to go back and work with their untalented, poorly trained managers who are just trying to get through the summer.best thing is, I think it might be a good idea for some of these employees to try to find another job. If they don't, maybe it is time for"
PiLM-Controller	0.20	'I don't care to go back and work with their untalented, untidy, and unorganized, and in many cases just plain plain wrong policies that are causing problems with the federal government," said Rep. John Shimkus (R-Ill.) on Wednesday, the second day of the Republican health care conference here.'

Table 18: Outputs of Language detoxification.