Hidden Space Patching to Prevent Undesired Prompt Completions

Nathaniel Getachew

May 2026

1 Introduction

Ensuring the safety and trustworthiness of large language models (LLMs) is a central challenge in their deployment. Modern LLMs are capable of generating highly coherent and detailed text, but this capability also enables them to produce harmful or undesirable content, including toxic language, misinformation, or instructions for illegal activities. Despite extensive post-training alignment techniques and safety filtering mechanisms, recent work has shown that sufficiently complex or adversarial prompts, such as those discovered through red-teaming or jailbreaking, can circumvent these safeguards and elicit unsafe outputs.

A large body of research has focused on mitigating these risks through post-training alignment methods, including reinforcement learning from human feedback, constitutional training, and adversarial fine-tuning. However, this emphasis on broad semantic safety leaves open a more targeted question: what happens when the objective is to suppress model performance on a fixed, known set of prompts? While prior work has explored memorization, data redaction, and selective unlearning, relatively little attention has been given to controlling model behavior at the level of specific prompts or narrow distributions of inputs. This setting is particularly relevant in scenarios where a predefined set of undesirable queries is known in advance.

In this work, we study whether it is possible to selectively degrade the performance of a language model on a fixed set of prompts while preserving its general capabilities. Rather than modifying the model through additional training, we investigate interventions at the level of internal representations and output decoding. We explore this question in the context of diffusion-based language models, which differ from traditional autoregressive models by generating outputs through an iterative denoising process over partially corrupted sequences.

We make the following contributions:

- We formulate the problem of *targeted utility suppression*, where the goal is to selectively degrade model performance on a fixed set of prompts.

- We propose representation-level interventions based on projecting hidden states away from subspaces associated with these prompts.
- We empirically evaluate different methods for constructing such subspaces, including PCA and discriminative approaches, and analyze their effect on both targeted and general performance.

2 Related Works

Prior work on LLM safety has focused mainly on broad alignment and refusal behavior rather than prompt-specific suppression. InstructGPT [9] established reinforcement learning from human feedback as a practical route for improving instruction following while reducing harmful or toxic outputs, and Constitutional AI [1] extended this paradigm with principle-based self-critique and AI feedback to train more harmless assistants. In parallel, the evaluation literature has emphasized adversarial robustness at the level of semantic safety categories: HarmBench [7] standardizes automated red teaming across a wide range of harmful behaviors, while jailbreak work shows that aligned models can still be induced to answer unsafe queries using optimized adversarial suffixes and other attack prompts [4].

Closer to our setting are selective unlearning and representation-level control. Approximate unlearning methods such as [5] aim to erase narrow bodies of knowledge from pretrained models with limited collateral damage, while capability-targeted approaches such as selective pruning weaken a specific behavior while preserving overall utility. At the representation level, INLP [10] and LEACE [2] remove linearly decodable concept information.

3 Methods

3.1 Llada Architecture

LLaDA [8] is a diffusion-based language model that generates text by iteratively reconstructing masked tokens in a partially corrupted sequence. Given an input sequence x_t, where a subset of tokens have been masked or replaced, the model predicts token distributions at all positions simultaneously.

At its core, LLaDA uses a standard Transformer architecture. The input sequence is first mapped to token embeddings and processed through a stack of L Transformer blocks. Each block consists of a self-attention layer followed by a feedforward network, with residual connections and normalization applied throughout. These blocks produce contextualized representations for each token:

$$H = \text{Transformer}(x_t) \in \mathbb{R}^{T \times d},$$

where T is the sequence length and d is the hidden dimension.

The final hidden states are then normalized using a layer normalization operation:

$$\tilde{H} = \ln_f(H).$$

To produce token predictions, LLaDA applies a shared linear unembedding layer to each position:

$$Z = \tilde{H} W_{\text{ff}},$$

where $W_{\text{ff}} \in \mathbb{R}^{d \times |\mathcal{V}|}$ maps hidden states to logits over the vocabulary $\mathcal{V}$, and $Z \in \mathbb{R}^{T \times |\mathcal{V}|}$ contains the logits for all tokens.

We can equivalently express the model as a composition of a representation function and a linear readout. Let

$$H(x_t) = \text{LN}_f(\text{Transformer}(x_t)) \in \mathbb{R}^{T \times d}$$

where LN is the LayerNorm function denote the final hidden state representation of the input. Then the model's output can be written compactly as:

$$Z = H(x_t) W_{\text{ff}} \tag{1}$$

This formulation is intended to highlight the separation between representation learning and output decoding. In this study, we ask if intervening on the learned representation can reduce the models ability to properly decode forbidden prompts.

3.2 Subspace Estimation

In order to prevent the LM from answering forbidden prompts only, we proceed with the assumption that the hidden representations produced by the model share common directions in representation space that are associated with these attributes. Formally, let $H(x_t) \in \mathbb{R}^{T \times d}$ denote the sequence of hidden states produced by the Transformer prior to the final normalization layer. We assume that there exists a low-dimensional subspace $S \subseteq \mathbb{R}^d$ such that components of $H(x_t)$ lying in S encode information correlated with the forbidden prompt set.

Under this assumption, suppressing the model's ability to utilize this information reduces to removing the contribution of this subspace from the hidden representations. To achieve this, we construct a projection onto the orthogonal complement of S. Let $U \in \mathbb{R}^{d \times k}$ be an orthonormal basis for S, so that the projection onto S is given by $UU^\top$. The projection onto the orthogonal complement $S^\perp$ is then:

$$P_\perp = I - UU^\top \tag{2}$$

Applying this projection to hidden states removes all components lying in S:

$$h' = hP_\perp,$$

for any token-level hidden state $h \in \mathbb{R}^d$. This operation preserves information orthogonal to the forbidden subspace while eliminating directions associated with the targeted attributes.

In practice, the subspace S is not known a-priori and may not even exist. We consider several methods for estimating this subspace, which are discussed in the next sections.

This formulation enables a simple intervention at the level of the output layer. By replacing the original readout matrix W_{ff} with

$$W'_{\text{ff}} = P_{\perp} W_{\text{ff}},$$

we obtain modified logits:

$$Z' = H(x_t) W'_{\text{ff}} = (H(x_t) P_{\perp}) W_{\text{ff}},$$

which depend only on components of the hidden representation orthogonal to the estimated forbidden subspace.

3.3 PCA Patching

Our first method of estimating the subspace S and its bases is via Principal Component Analysis. Principal Component Analysis (PCA) identifies a low-dimensional subspace that captures the dominant variation in a dataset. Given centered data $X \in \mathbb{R}^{N \times d}$, PCA computes the eigenvectors of the covariance matrix and selects the top k directions corresponding to the largest eigenvalues. In this study, we collect hidden states from the forbidden prompt set into a dataset $FP \in \mathbb{R}^{N \times d_{model}}$. We then conduct PCA across varying values of k to get k principal component vectors, which become our approximation for the orthonormal bases in Equation 2.

This method assumes that the directions of greatest variance across the forbidden prompt set's hidden states correspond to the directions in the hidden state space that set these prompts apart from other text. This is a strong assumption, and its validity is evaluated in Section 4.

3.4 Mean-Diff Patching

Our second method of estimating the subspace S and its bases is far simpler. We similarly construct a dataset of hidden states FP as above, but we also construct a dataset of hidden states $BP \in \mathbb{R}^{N \times d_{model}}$ of benign prompts, in this case prompts from the GSM8k dataset. We then compute the mean hidden state vectors for each of these, μ_F and μ_B respectively. In order to estimate the bases in Equation 2, we find the difference of the two means

$$\mu_* = \mu_F - \mu_B \tag{3}$$

and use this as our bases. This assumes that the hidden states will have the largest differences in the directions that matter the most to the forbidden prompt set.

4 Results

The goal of this study is to evaluate the two methods described above and see which of these reduces utility on a forbidden set of prompts the most, while preserving the ability to perform other tasks. In the following experiments, we use AllenAI's Toxic Prompts [6] dataset as our forbidden prompts. In the MeanDiff method, we use OpenAI's GSM8K dataset [3] as the benign dataset to pull from. The most appropriate metric would be to qualitatively asses model completions with and without patching. However due to lack of GPU access and time limits, we had to consider other metrics.

The first metric we consider is loss over the forbidden prompts. Ideally, we'd see a sharp increase in loss after patching, indicating that the model has reduced signal on this dataset. The second metric is the rank of each gold label token. Since our goal is to reduce the models ability to complete forbidden prompts verbatim, it's important that the rank of each token in the models probability distribution decreases after patching.

4.1 MeanDiff Reduces Harmful Prompt Utility More

As shown in Figure 1, both methods leave loss unchanged or actually reduce loss on the forbidden dataset. However, while both increase the average rank of the correct token, PCA increases it more aggressively. Coupled with the fact that PCA reduces loss by more as k increases, this may indicate that projecting out the principal components changes the models probability distribution in a way makes the literal gold token less likely, but related tokens more likely and unrelated tokens less likely, therefore increasing loss and decreasing gold token rank.

4.2 PCA Patching Worsens Benign Prompt Utility

The second requirement for the ideal patching method is to minimize loss in utility on unrelated prompts. Here we evaluate the model on the Books dataset, completely unrelated to the harmful prompts. Figure 2 shows that similar to before, PCA patching lowers loss on the books dataset, while MeanDiff leaves it relatively unaffected. However, we unfortunately see that PCA Patching also increases the mean rank of the gold tokens by nearly the same rate as it does on the harmful dataset. MeanDiff does increase the mean rank of the gold tokens in this setting, however it does so by less than it does in the harmful prompt setting. These results indicate the MeanDiff is likely a more targeted approach to preventing LMs from answering forbidden prompts.

Figure 3 plots the first two principal components of the hidden states across all three of the datasets we consider. Clearly, the hidden states are not easily seperable via PCA components.

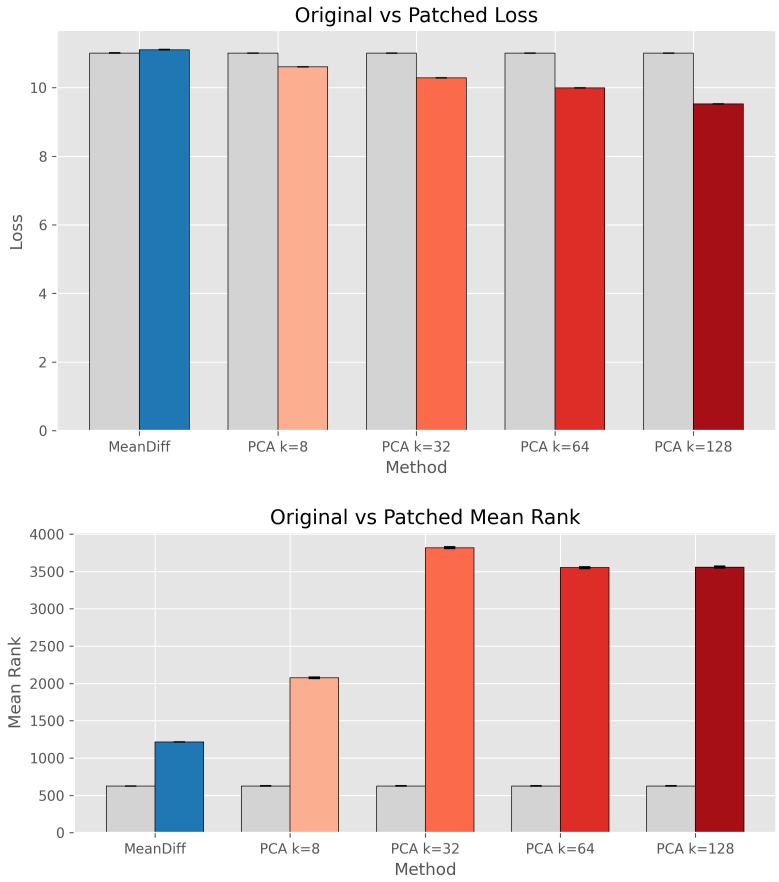

Figure 1: (**Top**) Loss on the AllenAI Toxicity prompt dataset across each method. In this case, higher values are better. (**Bottom**) Mean rank of the correct token across all tokens for each method. In this case, lower is better.

5 Conclusion

In this study, we propose two different methods for preventing a diffusion language model from correctly answering harmful prompts. PCA Patching displays the same behavior across all settings, while MeanDiff provides a more targetted solution, degrading performance more in the forbidden prompt setting than with benign prompts. These methods rely on heavy assumptions about the subspace that harmful prompts are encoded in, which is a severe limitation. Future work could explore this, and find better ways of estimating the subspace discussed in Equation 2, and further taking advantage of having an explicit set of forbidden prompts.

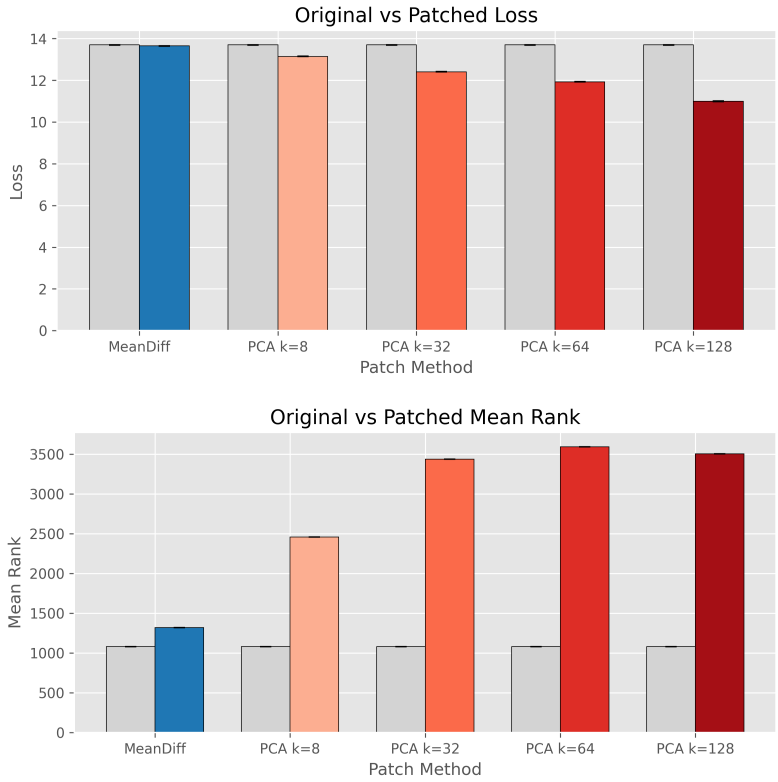

Figure 2: (**Top**) Loss on the Books prompt dataset across each method. In this case, lower values are better. (**Bottom**) Mean rank of the correct token across all tokens for each method. In this case, higher is better.

References

[1] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.

[2] Nora Belrose, Igor Ostrovsky, Lev McKinney, Zach Furman, Logan Smith, Danny Halawi, Stella Biderman, and Jacob Steinhardt. Eliciting latent predictions from transformers with the tuned lens. *arXiv preprint arXiv:2303.08112*, 2023.

[3] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

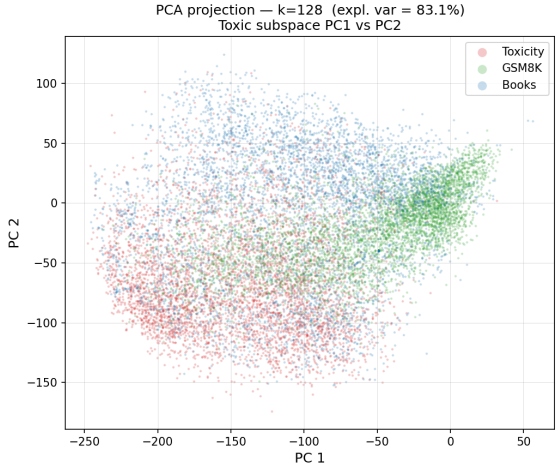

Figure 3: PCA Components for $k = 128$ across the three datasets considered in this dataset

[4] Tiehan Cui, Yanxu Mao, Peipei Liu, Congying Liu, and Datao You. Exploring jailbreak attacks on llms through intent concealment and diversion. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 20754–20768, 2025.

[5] Ronen Eldan and Mark Russinovich. Who's harry potter? approximate unlearning for llms. 2023.

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

[7] Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, et al. Harmbench: A standardized evaluation framework for automated red teaming and robust refusal. *arXiv preprint arXiv:2402.04249*, 2024.

[8] Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. *arXiv preprint arXiv:2502.09992*, 2025.

[9] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[10] Shauli Ravfogel, Michael Twiton, Yoav Goldberg, and Ryan D Cotterell. Linear adversarial concept erasure. In *International Conference on Machine Learning*, pages 18400–18421. PMLR, 2022.