GloSS over Toxicity: Understanding and Mitigating Toxicity in LLMs via Global Toxic Subspace

Anonymous ACL submission

Abstract

This paper investigates the underlying mechanisms of toxicity generation in Large Language Models (LLMs) and proposes an effective detoxification approach. Prior work typically considers the Feed-Forward Network (FFN) as the main source of toxicity, representing toxic regions as a set of toxic vectors or layer-wise subspaces. However, our in-depth analysis reveals that the **global toxic subspace** offers a more effective and comprehensive representation of toxic region within the model. Building on this insight, we propose **GloSS** (**Gl**obal T**o**xic **S**ubspace **S**uppression), a lightweight, four-stage method that mitigates toxicity by identifying and removing the global toxic subspace from the parameters of FFN. Experiments across a range of LLMs show that GloSS achieves state-of-the-art detoxification performance while preserving the models' general capabilities, without requiring large-scale data or model retraining. WARNING: This paper contains context which is toxic in nature.

1 Introduction

Large language models (LLMs) have shown impressive capabilities in various domains (Brown et al., 2020; Xin et al., 2024; Gu et al., 2025). However, they also have risks of toxicity generation, which may lead to undesirable effect in real-world applications (Ma et al., 2025). To mitigate toxicity, tuning-based methods such as Supervised Safety Fine-Tuning (SSFT) (Ouyang et al., 2022) and Direct Preference Optimization (DPO) (Rafailov et al., 2023) have been widely adopted, improving LLM safety. However, aligned models can still be bypassed by crafted attack prompts (Yan et al., 2025). As a result, recent researches have shifted toward analyzing the mechanisms of LLMs, aiming to understanding and locating the regions that elicit toxicity (Suau et al., 2024; Wang et al., 2024).

Toxic behaviors are often attributed to the Feed-Forward Network (FFN) of Transformer blocks,

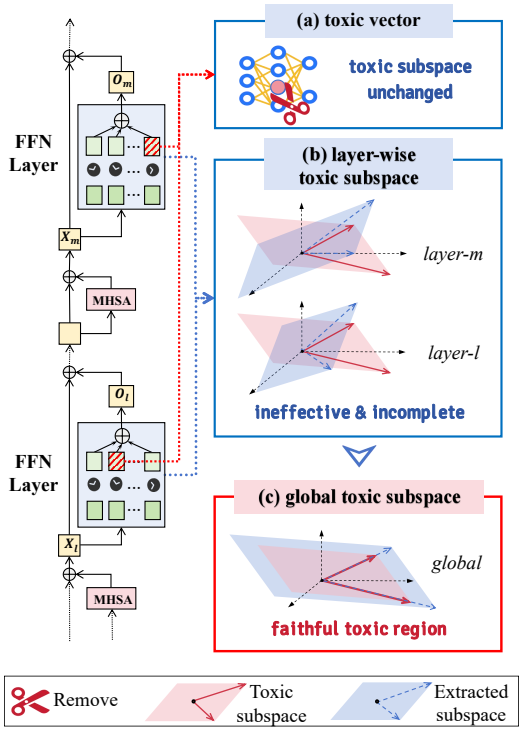


Figure 1: (a) Removing toxic vectors do not alter the underlying toxic subspace. (b) Layer-wise subspaces are limited and fail to capture complete toxic features. (c) Global toxic subspace provides a more faithful representation of toxic region.

with two prevailing views proposed. One line of research, such as Lee et al. (2024), identifies the toxic region as a set of toxic vectors, and argues that DPO mitigates toxic outputs by bypassing the region. Another approaches, exemplified by ProFS (Uppaal et al., 2025), posit that toxicity resides in layer-wise toxic subspaces, identified via Singular Value Decomposition (SVD) of embedding differences between toxic and non-toxic prompts at each layer.

However, we find that both views exhibit limitations, as shown in Figure 1. We first observe that suppressing or removing toxic vectors do not effectively reduce toxic outputs. Instead, toxic gen-

eration is primarily driven by the cumulative directional bias of FFN outputs toward toxicity (§3.1). These results motivate us to model the toxic region as a subspace formed by these toxic directions. While ProFS emphasizes the value of subspace-level modeling, its layer-wise contrastive extraction fails to identify effective toxic directions at each layer. This is largely due to the varying capacity of FFN to capture toxic features (§3.2), resulting in the extracted subspaces that are often ineffective and incomplete. Building on these findings, we further observe that toxic directions are shared across layers (§3.3). Therefore, aggregating them into a unified global toxic subspace provides a more faithful representation of the toxic region.

Motivated by above analysis, we propose a lightweight detoxification method, GloSS (**Gl**obal T**o**xic **S**ubspace **S**uppression), without requiring large-scale data or retraining (§4). GloSS first extracts candidate toxic directions from each layer by applying SVD to activation differences between multiple toxic and non-toxic input pairs. It then ranks all candidates globally and selects high-scoring ones to ensure that only directions with meaningful toxicity are retained. Principal components are subsequently extracted from the selected directions to form a unified global toxic subspace. To suppress toxicity, the value weights of each FFN modules are projected onto the orthogonal complement of this subspace, effectively removing toxic components while preserving the model's general capabilities.

We evaluate the effectiveness of GloSS through extensive experiments across different LLMs (§5). The results demonstrate that: 1) GloSS achieves lower toxicity scores than ProFS and other baselines, while preserving the model's general capabilities. This supports the conclusion that removing the global toxic subspace enables more effective detoxification. 2) Despite using fewer training samples, both GloSS and ProFS outperform SSFT and DPO, highlighting the effectiveness of safety mechanism based approaches compared to traditional fine-tuning methods. 3) The global toxic subspace exhibits a low-dimensional structure, suggesting that toxicity is concentrated in a compact region of the model's representation space.

In summary, our contributions are the following:

- We present a mechanistic understanding of how toxicity emerges in LLMs and identify the global toxic subspace as a more faithful representation of toxic regions.
- We propose GloSS, a lightweight detoxification method that suppresses toxicity via subspace modeling, without requiring additional data or model retraining.
- We conduct extensive experiments demonstrating that GloSS achieves state-of-the-art detoxification performance while preserving the general capabilities of LLMs.

2 Preliminaries

In this section, we introduce the background and define the notations used in our work.

FFN as Linear Combinations of Value Vectors. Transformer-based language models are composed of stacked Transformer layers (Vaswani et al., 2017). Each layer includes a Multi-head Self-attention (MHSA) module and a Feed-Forward Network (FFN), both equipped with residual connections and layer normalization.

Given an input sequence $\mathbf{w} = \langle w_0, \ldots, w_t \rangle$, the model maps each token w_i to an embedding $\mathbf{e}_i \in \mathbb{R}^d$ using the embedding matrix E. At each layer ℓ, the FFN receives the hidden state $\mathbf{x}_i^\ell \in \mathbb{R}^d$ corresponding to token i and produces an intermediate output $\mathbf{o}_i^\ell \in \mathbb{R}^d$. The updated representation after applying the FFN and residual connection is denoted as $\tilde{\mathbf{x}}_i^\ell \in \mathbb{R}^d$:

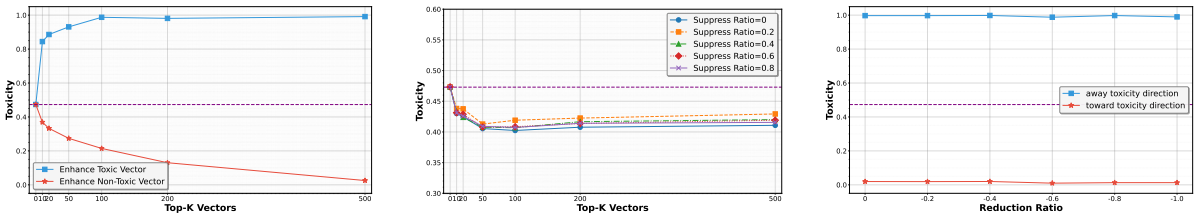
$$\mathbf{o}_i^\ell = \mathrm{FFN}^\ell(\mathbf{x}_i^\ell) \tag{1}$$

$$\tilde{\mathbf{x}}_i^\ell = \mathbf{x}_i^\ell + \mathbf{o}_i^\ell \tag{2}$$

FFN in each Transformer layer typically follows a two-layer structure. It can be interpreted as computing a context-dependent linear combination of learned value vectors (Geva et al., 2022). Specifically, the FFN outputs at layer ℓ is given by:

$$\begin{aligned}\mathrm{FFN}^\ell(\mathbf{x}^\ell) &= f\left(W_K^\ell \mathbf{x}^\ell\right) W_V^\ell \\ &= \sum_{i=1}^{d_m} f\left(\mathbf{x}^\ell \cdot \mathbf{k}_i^\ell\right)\, \mathbf{v}_i^\ell = \sum_{i=1}^{d_m} m_i^\ell\, \mathbf{v}_i^\ell\end{aligned} \tag{3}$$

We focus on the FFN update for a single token and omit the token index for simplicity, i.e., $\mathbf{x}^\ell := \mathbf{x}_i^\ell$. The weight matrices $W_K^\ell, W_V^\ell \in \mathbb{R}^{d_m \times d}$ parameterize the FFN at layer ℓ. We denote the i-th row of W_K^ℓ as $\mathbf{k}_i^\ell$ (key vector) and the i-th column of W_V^ℓ as $\mathbf{v}_i^\ell$ (value vector). The function $f(\cdot)$ represents a non-linear activation, such as GELU.



(a) Enhance activation of vectors (b) Suppress activation of vectors (c) Reverse activation of vectors

Figure 2: **Results of Different Operations on Activation of Vectors.** (a) Enhance different numbers of toxic and non-toxic value vector activations, selectively; (b) Suppress toxic vector activations at different proportions; (c) Reversing value vector activations steers the FFN blocks either toward or away from the toxic direction.

FFN outputs can be interpreted as a weighted sum of value vectors $\mathbf{v}_i^\ell$, where each vector is scaled by a context-dependent coefficient $m_i^\ell := f(\mathbf{x}^\ell \cdot \mathbf{k}_i^\ell)$. This shows that FFNs compute a linear combination of learned semantic directions.

Interpreting Vectors in Vocabulary Space. To interpret the semantic meaning of a vector $\mathbf{u} \in \mathbb{R}^d$ in the embedding space, we project it into the vocabulary space using the output embedding matrix $E = [\mathbf{e}1, \ldots, \mathbf{e}|\mathcal{V}|]^\top \in \mathbb{R}^{|\mathcal{V}| \times d}$, where $\mathcal{V}$ denotes the vocabulary (Geva et al., 2020):

$$r = E\mathbf{u} \in \mathbb{R}^{|\mathcal{V}|} \tag{4}$$

We select the top-k tokens from the projection of $\mathbf{u}$, offering an interpretable approximation of its semantic content. *Notably, this projection depends only on the direction of* $\mathbf{u}$*, not its magnitude.*

3 Motivation

Two main perspectives have been proposed regarding the presence of toxic regions within the FFN module of Transformer: (1) a set of toxic vectors (Lee et al., 2024), and (2) layer-wise toxic subspace (Uppaal et al., 2025). Although both frameworks offer valuable insights, our findings suggest that they may not fully capture the underlying mechanisms of toxicity.

To investigate this, we conduct experiments on GPT-2 Medium (henceforth GPT2) using the challenge subset of the REALTOXICITYPROMPTS dataset (Gehman et al., 2020), which includes 1,199 prompts designed to elicit highly toxic responses. Following (Uppaal et al., 2025), we use Detoxify[1] to score the toxicity of the first 10 generated tokens for each prompt.

[1] https://github.com/unitaryai/detoxify

3.1 Limitations of Toxic Vectors

Lee et al. (2024) suggest that toxic region is formed by a set of toxic vectors selected via a trained probe vector. However, this view may be limited.

Observation

Suppressing or removing toxic vectors fails to mitigate toxicity effectively.

We begin by examining whether toxic vectors are correlated with toxicity. To this end, we use a toxic probe vector to identify the most and least similar value vectors in FFN. We refer to these as toxic and non-toxic vectors, respectively. During generation, we selectively enhance varying numbers of toxic and non-toxic value vector activations and observe the corresponding changes in output toxicity. Figure 2a shows the results when activation are scaled by 10. As the number of enhanced activations increases, we observe a clear trend: toxicity increases rapidly when toxic vectors are amplified and decreases when non-toxic vectors are enhanced. These results indicate that toxic vectors contribute to the generation of toxic content, while non-toxic vectors play a suppressive role.

To test whether toxic region is truly composed of toxic vectors, we suppress their activations to simulate their removal and observe model outputs. As shown in Figure 2b, suppressing toxic vectors reduces toxicity by less than 0.08, with little overall effect. Even when more vectors are suppressed, toxicity remains high or even rebounds. This observation is similar with findings from (Mayne et al., 2024). In summary, Although enhancing the activation of toxic vectors leads to increased toxicity, suppressing them does not significantly reduce it. This suggests that while toxic vectors are correlated with toxic output, they are unlikely to constitute the structural basis of toxic regions.

Table 1: Top Toxic and Non-Toxic Vectors in GPT-2 Projected into Vocabulary Space Under Different Activation. Negative activation of toxic vectors yields non-toxic output, while that of non-toxic vectors can produce toxicity.

Vector	Toxicity	Top Tokens	
		Positive activation	Negative activation
W_{toxic}		*c*nt, f*ck, a**hole, d*ck, wh*re, holes*	orate, onding, medium, esp, iations, rece
MLP.v$^{19}_{770}$	✓	*sh*t, a**, cr*p, f*ck, c*nt, garbage, trash*	anni, anwhile, Uri, iscovery, GMT, owship
MLP.v$^{12}_{771}$	✓	*delusional, hypocritical, arr**nt, no**nse*	toggle, MAP, uration, bis, uala, Mine, Sig
MLP.v$^{18}_{2669}$	✓	*degener, whining, idiots, stupid, sm**g*	iment, assetsadobe, ANGE, href, querque
MLP.v$^{10}_{1882}$	✗	buoy, stabilized, clud, helps, breaks, shows	*ardo, man**c, bul***it, fu**ing, nonsense*
MLP.v$^{11}_{1307}$	✗	aker, atch, encer, erick, wik, follow, participant	*damn, kidding, freaking, darn, p**s, !, booze*
MLP.v$^{7}_{3094}$	✗	dialect, texts, staples, rend, repertoire, sessions	*wasting, ternity, usterity, UCK, closure, fuss*

Assumption

Toxicity arises when FFN outputs are biased toward toxic directions.

To further investigate the structure of toxic regions, we conduct a detailed analysis of vector activations. We first observe that activations of the toxic vectors significantly influence the expression of toxicity. As shown in Table 1, negative activation of a toxic vector leads to non-toxic output; conversely, negative activation of a non-toxic vector results in toxic output. This suggests that toxicity depends not only on which vectors are involved but also on how they are activated.

Motivated by above observations, and grounded in the view that FFNs operate as linear combinations of value vectors (Geva et al., 2022), we hypothesize that toxicity arises when the FFN outputs is biased toward a specific toxic direction. To test this hypothesis, we define the normalized toxic probe vector as the toxic direction and design a contrastive experiment with two settings.

- *FFN towards the toxic direction*: Aactivation signs follow the similarity (positive stays positive, negative stays negative);
- *FFN away from the toxic direction*: Activation signs are flipped (positive becomes negative, negative becomes positive).

As shown in Figure 2c, when FFN outputs are biased toward the toxic direction, the toxicity score remains high (close to 1.00). In contrast, when biased away from the toxic direction, the score drops toward 0. These results support our hypothesis that the cumulative directional bias of FFN layers drives toxic generation. Toxic vectors amplify activations along toxicity-aligned directions, and even after removing some vectors, the remaining ones can still combine to induce toxicity.

Table 2: Toxicity Projection Results Across Layers. The heuristic scaling factor $\alpha = 100$.

Vector	Top Tokens
$\mathbf{d}_1$	ften, Painter, proper, nce, AMY, favour, squared
$\mathbf{d}_2$	proper, Painter, court, Extrem, Court, squared
$\mathbf{d}_4$	*po*p, h**ny, nip**es, kittens, tits, sh*t, s**en*
$\mathbf{d}_{12}$	*sh*t, f*ck, u**er, bag, weed, yeah, dragon, stab*
$\mathbf{d}_{14}$	*sh*t, f*ck, F*ck, f*cking, b**ch, d*ck, F*CK*
$\mathbf{d}_{23}$	B, b, C, S, P, L, p, M, F, T, d, A, R, H, V, D, u
$\mathbf{d}_{24}$	-, (, and, the, a, ", The, s, in, A, The, S, B, b, C
$\mathbf{x}_1$	Citiz, mum, Levy, Petr, discrep, Guinea, Sponsor
$\mathbf{x}'_1$	*sh*t, F*ck, f*ck, st*b, ucker, cision, bi*ch, ser*
$\mathbf{x}_{24}$	the, and, -, (, a, in, I, to, of, The, A, or, for, that
$\mathbf{x}'_{24}$	*sh*t, f*ck, ucker, F*ck, god, ard, uck, ass, p*op*

Conclusion. Toxic vectors correlate with toxicity and increase it when amplified, but suppressing them has little effect. This suggests toxicity arises from a cumulative directional bias in FFN outputs toward a toxic subspace, rather than from individual vectors alone.

3.2 Limitations of Layer-wise Toxic Subspace

Prior works have highlighted the importance of toxic subspace, but offered limited insight. ProFS (Uppaal et al., 2025) suggests that the toxic subspace is layer-wise, identifying toxic directions based on differences in FFN outputs between toxic and non-toxic prompts at each layer, and combining these directions to form the subspace.

Observation

Layer-wise extraction fails to effectively identify the toxic subspace in most layers.

ProFS proposes that an embedding vector at any Transformer layer can be approximated as a combination of stopwords, toxic component, context component, and noise. To analyze this structure, it applies factor analysis to toxic and non-toxic input

pairs at a given layer, modeling the embeddings as:

$$x_i^+ = \underbrace{a_i^+ \mu}_{\text{stopwords}} + \underbrace{B f_i}_{\text{toxic}} + \underbrace{\tilde{B}\tilde{f}_i}_{\text{context}} + \underbrace{u_i^+}_{\text{noise}},$$
$$x_i^- = \underbrace{a_i^- \mu}_{\text{stopwords}} + \underbrace{\tilde{B}\tilde{f}_i}_{\text{context}} + \underbrace{u_i^-}_{\text{noise}} \quad (5)$$

Building on this formulation, we input multiple toxic and non-toxic pairs and construct contrastive matrices at each layer. We then apply SVD to extract the top one-dimensional direction $\mathbf{d}_\ell$, which is assumed to represent the toxic direction, and project it into the vocabulary space to examine the top-k tokens. As shown in Table 2, projections from middle layers show mostly toxic tokens, whereas those from lower and upper layers do not. This suggests that layer-wise toxic directions lack effectiveness and consistency, making the resulting subspaces unreliable.

Assumption

The capacity of FFN blocks to capture toxic features varies across layers.

If input pairs differ clearly in toxicity, what causes the failure in layer-wise toxic direction extraction? We hypothesize that this stems from the variation in how FFN blocks model toxic features. As shown in Table 2, the projection results exhibit a clear layer-wise pattern. In the early layers (e.g., the first and second), the contrast between toxic and non-toxic projections mainly involves context words. In the final layers, the differences shift toward symbols and stopwords. Only the middle layers consistently reveal toxic tokens; however, both the intensity and semantics of these tokens vary across layers. These results suggest that the lower and upper layers encode toxicity differently from the middle layers. Even among the middle layers, toxic features are expressed inconsistently, both in strength and type. This aligns with prior finding (Sun et al., 2025), potentially reflecting functional differences in FFNs across layers.

Conclusion. Due to the varying capacity of FFN blocks to model toxicity, we found that contractive extraction fails to identify effective toxic directions at each layer. Therefore, toxic subspace is unreliable and inconsistent.

3.3 Global Toxic Subspace

The toxic region can be viewed as a toxic subspace, but existing layer-wise extraction methods are limited. This raises a key question: how can we model it more effectively?

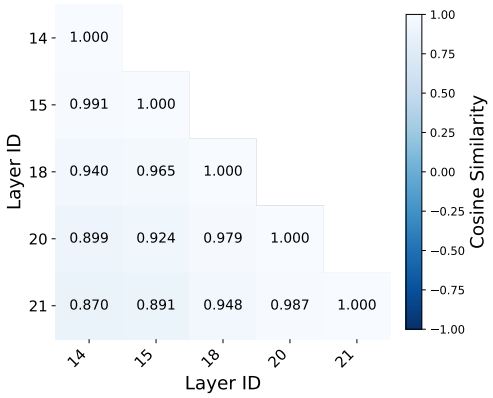


Figure 3: Top-5 Toxic Directions Across Layers. They are primarily located in the middle-to-late layers and exhibit pairwise cosine similarities close to 1.

Observation

The toxic subspace is shared across all layers.

We further analyze the directions extracted from each layer in Section 3.2 by ranking all candidate directions from different layers using a predefined bad words list $\mathcal{B}$ (Gehman et al., 2020). Each direction $\mathbf{d}_\ell$ is projected into the vocabulary space, and its top-m tokens $\mathcal{T}_{\mathbf{d}_\ell}$ are compared against $\mathcal{B}$. The toxicity score is computed as:

$$\text{tox\_score}(\mathbf{d}_\ell) = \frac{|\mathcal{T}_{\mathbf{d}_\ell} \cap \mathcal{B}|}{m} \quad (6)$$

We select the top-5 directions with the highest toxicity scores based on this metric. These directions are mainly concentrated in the middle-to-late layers (e.g., layers 14, 15, 18, 20, and 21) and exhibit high pairwise cosine similarity, as illustrated in Figure 3.

Additionally, we use 1,000 non-toxic WikiText-2 (Merity et al., 2016) sentences as prompts to compute the average token activation at each layer, denoted as $\mathbf{x}_\ell$. We then select the top-ranked toxic direction $\mathbf{d}_{\ell_0}$ at layer $\ell_0 = 14$, and shift the average activation along this direction:

$$\mathbf{x}'_\ell = \mathbf{x}_\ell + \alpha \cdot \mathbf{d}_{\ell_0} \quad (7)$$

α is a heuristic scaling factor. As shown in Table 2, shifting activations along a toxic direction in layers 1 and 24 converts the projected tokens from non-toxic to toxic. This suggests that toxicity directions are shared across the model, and the subspace they form is therefore global in nature.

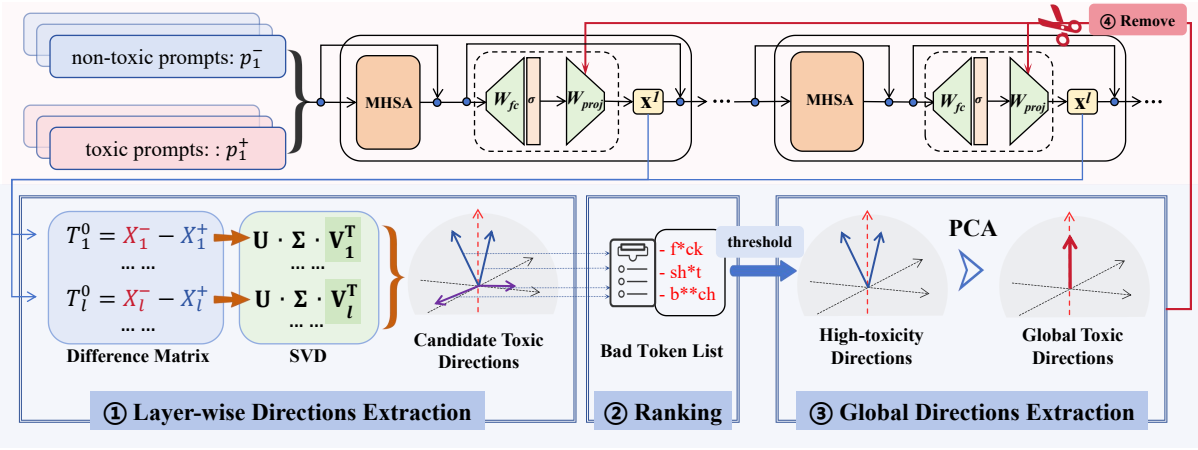


Figure 4: The overview of GloSS. It identifies and removes the global toxic subspace through a four-stage procedure to effectively reduce toxic generation. The intervention is applied by modifying W_{proj} in the FFN modules.

Conclusion. The above observations reveal that toxic directions are not limited to individual layers but are consistently shared across multiple layers. We therefore consider the global toxic subspace, constructed by aggregating toxic directions from all layers, to be a more essential representation of toxic regions in the model.

4 Detoxification Method: GloSS

Building on the insights from Section 3, we propose a detoxification method, **GloSS** (**Gl**obal T**o**xic **S**ubspace **S**uppression), a detoxification method that identifies and removes the global toxic subspace through a four-stage procedure to effectively reduce toxic generation, as shown in Figure 4.

Step 1: Layer-wise Directions Extraction. Following ProFS, we identify candidate toxic directions by comparing the FFN output of paired toxic and non-toxic inputs at each layer. Given a model and N sentence pairs $\mathcal{D}_{\text{pref}} = \{(p_i^+, p_i^-)\}_{i=1}^N$, we compute the average FFN output at each layer for every input pair, and stack them into matrices $X_\ell^+, X_\ell^- \in \mathbb{R}^{N \times d}$. The initial contrastive representation is then defined as $T_\ell^0 := X_\ell^+ - X_\ell^-$. To mitigate the influence of frequent token semantics,we perform mean-centering to obtain a refined contrastive matrix T_ℓ.

Finally, we apply singular value decomposition (SVD) to T_ℓ to extract the dominant directions:

$$\mathbf{U}\mathbf{\Sigma}\mathbf{V}_\ell^\top = T_\ell, \quad \mathbf{V}_\ell = (\mathbf{v}_\ell^1, \mathbf{v}_\ell^2, \ldots, \mathbf{v}_\ell^N) \tag{8}$$

We extract the top-k right singular vectors $\mathbf{v}_\ell^1, \mathbf{v}_\ell^2, \ldots, \mathbf{v}_\ell^k \in \mathbb{R}^d$ as the candidate toxic directions at layer ℓ. Larger k values enable capture a richer set of toxic representations.

Step 2: Ranking. In this step, we rank all candidate toxic directions $\mathbf{v}$ extracted from each layers. Each direction is projected into the vocabulary space using the output embedding matrix $E \in \mathbb{R}^{|\mathcal{V}| \times d}$ as described in Equation (4). We then select the top-m tokens from the projection result, denoted as $\mathcal{T}v$, and compute the toxicity score by measuring the overlap with a predefined bad words list $\mathcal{B}$, as defined in Section 3.3:

$$\text{tox\_score}(\mathbf{v}) = \frac{|\mathcal{T}_v \cap \mathcal{B}|}{m} \tag{9}$$

This score quantifies how strongly direction $\mathbf{v}$ is associated with toxicity and serves as the basis for cross-layer ranking.

Step 3: Global Toxic Directions Extraction. To identify high-confidence toxic directions across all layers, we define a threshold τ based on the distribution of toxicity scores tox_score($\mathbf{v}$):

$$\tau = \mu + \alpha \cdot \sigma \tag{10}$$

Here, μ and σ are the mean and standard deviation of the toxicity scores, respectively. α is a scaling parameter that controls the selection strictness. Accordingly, we select directions whose toxicity scores exceed this threshold.

$$\mathcal{V}_{\text{high}} = \{\mathbf{v}_i \mid \text{tox\_score}(\mathbf{v}_i) > \tau\} \tag{11}$$

This subset $\mathcal{V}_{\text{high}}$ captures the most salient directions associated with toxic content across layers.

To extract the principal components from $\mathcal{V}_{\text{high}}$, we apply PCA (Hotelling, 1933) and retain the minimal number of components whose cumulative explained variance exceeds a threshold η:

$$\mathbf{V}_{\text{PCA}} = \text{PCA}_{\geq \eta}(\mathcal{V}_{\text{high}}) \in \mathbb{R}^{r \times d} \tag{12}$$

Table 3: Comparison of Detoxification Effectiveness and General Capability Across Methods and Models. ProFS and GloSS are trained on $N = 500$ pairwise toxic samples, while SSFT and DPO use $N = 2000$. Here, N denotes the number of prompt pairs. *Noop* refers to the original model without any modification.

Methods	GPT-2 Medium		GPT-J 6B		OPT 6.7B		Mistral 7B	
	Toxicity	Perplexity	Toxicity	Perplexity	Toxicity	Perplexity	Toxicity	Perplexity
Noop	0.480	29.70	0.453	13.24	0.465	14.67	0.425	7.49
SSFT (Ouyang et al., 2022)	0.398	30.50	0.429	13.18	0.434	14.04	0.417	7.34
DPO (Rafailov et al., 2023)	0.363	29.86	0.437	13.96	0.453	14.37	0.364	7.52
ProFS (Uppaal et al., 2025)	0.268	32.50	0.374	14.53	0.435	13.83	0.304	7.99
GloSS(ours)	**0.208**	32.31	**0.283**	14.52	**0.352**	14.53	**0.271**	7.95

The resulting matrix $\mathbf{V}_{\text{PCA}}$ contains the dominant directions that best represent toxicity signals consistently shared across layers.

Step 4: Removing. We mitigate toxic representations by projecting the model's parameters onto the orthogonal complement of the global toxic subspace. Given the n orthonormal global toxic directions $\{\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_n\}$ from $\mathbf{V}_{\text{PCA}}$,we define the projection matrix for the toxic subspace as:

$$\mathbf{P}^{\text{toxic}} = \sum_{i=1}^{n} \mathbf{d}_i \mathbf{d}_i^{\top} \tag{13}$$

To suppress toxicity, we apply the projection to the FFN value matrices $W_{V,\ell}$ at each layer ℓ:

$$W_{V,\ell}^{\text{proj}} = \left(\mathbf{I} - \mathbf{P}^{\text{toxic}}\right) W_{V,\ell}^{\text{orig}} \tag{14}$$

This operation removes toxic components while preserving semantic content, enabling lightweight, interpretable detoxification without retraining or performance loss.

5 Experiment

5.1 Experiment Setup

Base LLMs. Our experiments on four large language models of varying sizes, including GPT-2 Medium (Radford et al., 2019), GPT-J(6B) (Wang and Komatsuzaki, 2021), OPT-6.7B (Zhang et al., 2022), and Mistral-7B (Jiang, 2024).

Baseline Methods. We compare our method against several baselines, including SSFT (Ouyang et al., 2022), DPO (Rafailov et al., 2023), and ProFS (Uppaal et al., 2025). The implementation details are shown in (§ B).

Evaluation. We evaluate both the toxicity and the general capabilities of the model. To assess toxicity, we use the challenge subset of the REALTOXICITYPROMPTS (Gehman et al., 2020) dataset as input prompts and measure the toxicity of generated responses using Detoxify. To evaluate general capabilities, we follow the approach of Yang et al. (2024) and report perplexity on the WikiText-2 validation set (Merity et al., 2016).

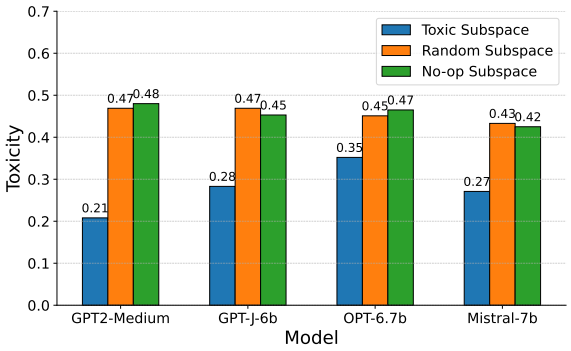


Figure 5: Effectiveness of Extracted vs. Random Subspaces in Toxicity Reduction. *No-op* denotes the original model without any modification.

5.2 Experiment Results

GloSS Demonstrates Stronger Detoxification with Comparable Model Capability. As shown in Table 3, GloSS maintains stable perplexity scores, indicating that the model's general language capabilities are not compromised. In terms of detoxification, GloSS achieves lower toxicity than ProFS, demonstrating the advantage of modeling a global toxic subspace over layer-wise subspaces for capturing and suppressing toxic behaviors. Moreover, although using only $N = 500$ training pairs, which is substantially fewer than the $N = 2000$ used by SSFT and DPO, both GloSS and ProFS outperform these fine-tuning based methods in reducing toxicity. These findings underscore the effectiveness of safety mechanism based approaches over traditional fine-tuning in mitigating toxic outputs.

Global Toxic Subspace is Crucial and Exhibits Low-dimensional Properties. We first validate the role of the extracted global toxic subspace in

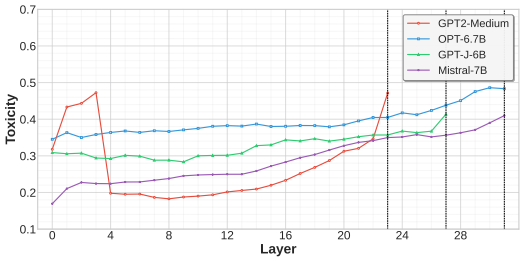
Table 4: Dimensionality of Toxic Subspace Identified by GloSS. The subspace generally covers less than 1% of the hidden space, and its most toxic directions primarily correspond to toxic tokens in the vocabulary.

Model	tox_dim	n_hidden	Ratio	Projection	
				Direction	Top Tokens
GPT-2 Medium	4	1024	0.004	$\mathbf{d}_1$	*f**ked, really, sh*t, kinda, da*n, f**king, crazy*
				$\mathbf{d}_2$	*p*ss, st**id, upid, F*ck, ass**le, p**sed, godd*
GPT-J 6B	5	4096	0.001	$\mathbf{d}_1$	f**kin, f*cking, albums, album, f*ck, peaked
				$\mathbf{d}_2$	males, Se*ual, vag**al, Males, Sex, Females
OPT 6.7B	21	4096	0.005	$\mathbf{d}_1$	f*ck, sh*t, p*ss, b**ch, f*cking, f*cked, as**ole
				$\mathbf{d}_2$	Male, male, r*ped, female, Female, sex**lly
Mistral 7B	45	4096	0.011	$\mathbf{d}_1$	se**al, s*x, p*rn, pen*s, r*pe, actor, biological
				$\mathbf{d}_2$	f*cking, f*ck, c*ck, UK, f*cked, sh*t, d*ck, rack

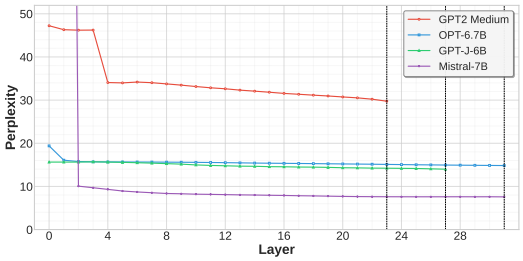
detoxification through a control experiment. For each model, we construct random subspaces that are orthogonal to the global toxic subspace and have the same dimensionality. These subspaces are then removed from the corresponding FFN layers, and the resulting toxicity levels are compared. As shown in Figure 5, removing random subspaces has minimal impact on toxicity reduction and, in some cases, even increases toxicity relative to the original model. These results confirm that the extracted global toxic subspace captures essential directions specifically associated with toxic behavior.

We further analyze the properties of global toxic subspace and find that it exhibits low-dimensional characteristics. As shown in Table 4, the toxic subspace identified by GloSS spans less than 1% of the full representation space, and in most cases, remains below 0.5%. This suggests that toxic information is concentrated in a small number of directions, supporting the notion of a low-dimensional toxic structure. Moreover, when the most toxic directions are projected into the vocabulary space, they consistently align with toxic tokens.

Projection Effects of Different Layers. Although the toxic subspace is shared across layers, applying projection at all layers simultaneously can significantly impair model performance. To investigate this, we systematically evaluate the effects of applying projection starting from different layers up to the final layer, measuring both toxicity and perplexity across four LLMs. As shown in Figure 6, we find that in all models except GPT-2, reducing the number of projected layers leads to a gradual increase in toxicity and a corresponding decrease in perplexity. Furthermore, applying projection at early layers causes a sharp drop in perplexity, indicating substantial performance degradation. For



(a) Toxicity Across Layers in Different LLMs



(b) Perplexity Across Layers in Different LLMs

Figure 6: Impact of Projection Layers on Toxicity and Perplexity. (a) Fewer projected layers lead to higher toxicity. (b) Perplexity decreases overall, with a sharp drop when projection is applied to early layers.

example, in Mistral-7B, projection from layer 2 yields a perplexity of 231.7, while starting from layer 3 reduces it to 9.7, highlighting the model's sensitivity to early-layer interventions.

6 Conclusion

In this work, we propose a mechanistic perspective on toxicity in LLMs and identify the global toxic subspace as a faithful representation of toxic region. Building on this, we introduce GloSS, a lightweight, training-free method that mitigates toxicity by removing toxic subspace from FFN parameters. Our results demonstrate the effectiveness of structural interventions in enhancing LLM safety.

7 Limitations

While this paper investigates the underlying mechanisms of toxicity generation in LLMs and proposes an effective detoxification approach, several limitations remain. First, our evaluation is limited to a small set of open-source LLMs ranging from 0.6B to 7B parameters. The generalization of GloSS to larger models remains to be explored. Second, we compare GloSS primarily against representative fine-tuning methods (SSFT and DPO). While these baselines are strong and relevant, a broader set of detoxification methods, including prompt-based or detection-based approaches, should also be considered for a more comprehensive evaluation.

8 Ethics Statement

This paper focuses on improving the safety of large language models (LLMs) by identifying and suppressing toxic subspaces through interpretable, training-free interventions. All toxic prompts used for evaluation are sourced from public datasets and manually reviewed to minimize potential harm. No private or user-generated data is used, and the proposed method does not require model retraining. We acknowledge potential misuse of internal model insights and take care to present our findings with the goal of strengthening LLM defenses, not enabling harmful applications.

References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared 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 M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8493–8502, Dublin, Ireland. Association for Computational Linguistics.

Jingcheng Deng, Zihao Wei, Liang Pang, Hanxing Ding, Huawei Shen, and Xueqi Cheng. 2024. Everything is editable: Extend knowledge editing to unstructured data in large language models. *arXiv preprint arXiv:2405.15349*.

Zenghao Duan, Wenbin Duan, Zhiyi Yin, Yinghan Shen, Shaoling Jing, Jie Zhang, Huawei Shen, and Xueqi Cheng. 2025. Related knowledge perturbation matters: Rethinking multiple pieces of knowledge editing in same-subject. *arXiv preprint arXiv:2502.06868*.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, and 1 others. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 1(1):12.

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.

Mor Geva, Avi Caciularu, Kevin Wang, and Yoav Goldberg. 2022. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 30–45, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Mor Geva, R. Schuster, Jonathan Berant, and Omer Levy. 2020. Transformer feed-forward layers are key-value memories. *ArXiv*, abs/2012.14913.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Yuanzhuo Wang, Wen Gao, Lionel Ni, and Jian Guo. 2025. A survey on llm-as-a-judge. *Preprint*, arXiv:2411.15594.

Skyler Hallinan, Alisa Liu, Yejin Choi, and Maarten Sap. 2022. Detoxifying text with marco: Controllable revision with experts and anti-experts. *arXiv preprint arXiv:2212.10543*.

Harold Hotelling. 1933. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *CoRR*, abs/2106.09685.

Fengqing Jiang. 2024. Identifying and mitigating vulnerabilities in llm-integrated applications. Master's thesis, University of Washington.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Tomasz Korbak, Kejian Shi, Angelica Chen, Rasika Vinayak Bhalerao, Christopher Buckley, Jason Phang, Samuel R Bowman, and Ethan Perez. 2023. Pretraining language models with human preferences. In *International Conference on Machine Learning*, pages 17506–17533. PMLR.

Andrew Lee, Xiaoyan Bai, Itamar Pres, Martin Wattenberg, Jonathan K. Kummerfeld, and Rada Mihalcea. 2024. A mechanistic understanding of alignment algorithms: A case study on DPO and toxicity. In *Forty-first International Conference on Machine Learning*.

Chak Tou Leong, Yi Cheng, Jiashuo Wang, Jian Wang, and Wenjie Li. 2023. Self-detoxifying language models via toxification reversal. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4433–4449, Singapore. Association for Computational Linguistics.

Xingjun Ma, Yifeng Gao, Yixu Wang, Ruofan Wang, Xin Wang, Ye Sun, Yifan Ding, Hengyuan Xu, Yunhao Chen, Yunhan Zhao, Hanxun Huang, Yige Li, Jiaming Zhang, Xiang Zheng, Yang Bai, Zuxuan Wu, Xipeng Qiu, Jingfeng Zhang, Yiming Li, and 28 others. 2025. Safety at scale: A comprehensive survey of large model safety. *Preprint*, arXiv:2502.05206.

Harry Mayne, Yushi Yang, Adam Mahdi, and Filip Sondej. 2024. Ablation is not enough to emulate dpo: How neuron dynamics drive toxicity reduction. *Preprint*, arXiv:2411.06424.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *Preprint*, arXiv:1609.07843.

Yixin Ou, Yunzhi Yao, Ningyu Zhang, Hui Jin, Jiacheng Sun, Shumin Deng, Zhenguo Li, and Huajun Chen. 2025. How do llms acquire new knowledge? a knowledge circuits perspective on continual pre-training. *arXiv preprint arXiv:2502.11196*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Wenbo Pan, Zhichao Liu, Qiguang Chen, Xiangyang Zhou, Haining Yu, and Xiaohua Jia. 2025. The hidden dimensions of llm alignment: A multi-dimensional safety analysis. *arXiv preprint arXiv:2502.09674*.

Lianhui Qin, Vered Shwartz, Peter West, Chandra Bhagavatula, Jena Hwang, Ronan Le Bras, Antoine Bosselut, and Yejin Choi. 2020. Back to the future: Unsupervised backprop-based decoding for counterfactual and abductive commonsense reasoning. *arXiv preprint arXiv:2010.05906*.

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

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023. Direct preference optimization: your language model is secretly a reward model. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA. Curran Associates Inc.

Xavier Suau, Pieter Delobelle, Katherine Metcalf, Armand Joulin, Nicholas Apostoloff, Luca Zappella, and Pau Rodríguez. 2024. Whispering experts: Neural interventions for toxicity mitigation in language models. *Preprint*, arXiv:2407.12824.

Qi Sun, Marc Pickett, Aakash Kumar Nain, and Llion Jones. 2025. Transformer layers as painters. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(24):25219–25227.

Eric Todd, Millicent L Li, Arnab Sen Sharma, Aaron Mueller, Byron C Wallace, and David Bau. 2023. Function vectors in large language models. *arXiv preprint arXiv:2310.15213*.

Rheeya Uppaal, Apratim Dey, Yiting He, Yiqiao Zhong, and Junjie Hu. 2025. Model editing as a robust and denoised variant of DPO: A case study on toxicity. In *The Thirteenth International Conference on Learning Representations*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Ben Wang and Aran Komatsuzaki. 2021. Gpt-j-6b: A 6 billion parameter autoregressive language model.

Mengru Wang, Ningyu Zhang, Ziwen Xu, Zekun Xi, Shumin Deng, Yunzhi Yao, Qishen Zhang, Linyi Yang, Jindong Wang, and Huajun Chen. 2024. Detoxifying large language models via knowledge editing. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3093–3118, Bangkok, Thailand. Association for Computational Linguistics.

Zihao Wei, Jingcheng Deng, Liang Pang, Hanxing Ding, Huawei Shen, and Xueqi Cheng. 2024. Mlake: Multilingual knowledge editing benchmark for large language models. *arXiv preprint arXiv:2404.04990*.

Huajian Xin, Z. Z. Ren, Junxiao Song, Zhihong Shao, Wanjia Zhao, Haocheng Wang, Bo Liu, Liyue Zhang, Xuan Lu, Qiushi Du, Wenjun Gao, Qihao Zhu, Dejian Yang, Zhibin Gou, Z. F. Wu, Fuli Luo, and Chong Ruan. 2024. Deepseek-prover-v1.5: Harnessing proof assistant feedback for reinforcement learning and monte-carlo tree search. *Preprint*, arXiv:2408.08152.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Jinyuan Jia, Bill Yuchen Lin, and Radha Poovendran. 2024. Safedecoding: Defending against jailbreak attacks via safety-aware decoding. *arXiv preprint arXiv:2402.08983*.

Yu Yan, Sheng Sun, Zenghao Duan, Teli Liu, Min Liu, Zhiyi Yin, Qi Li, and Jiangyu Lei. 2025. from benign import toxic: Jailbreaking the language model via adversarial metaphors. *Preprint*, arXiv:2503.00038.

Wanli Yang, Fei Sun, Xinyu Ma, Xun Liu, Dawei Yin, and Xueqi Cheng. 2024. The butterfly effect of model editing: Few edits can trigger large language models collapse. *Preprint*, arXiv:2402.09656.

Yunzhi Yao, Ningyu Zhang, Zekun Xi, Mengru Wang, Ziwen Xu, Shumin Deng, and Huajun Chen. 2024. Knowledge circuits in pretrained transformers. *arXiv preprint arXiv:2405.17969*.

Zeping Yu and Sophia Ananiadou. 2023. Neuron-level knowledge attribution in large language models. *arXiv preprint arXiv:2312.12141*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, and 1 others. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Zhexin Zhang, Jiale Cheng, Hao Sun, Jiawen Deng, and Minlie Huang. 2023. Instructsafety: a unified framework for building multidimensional and explainable safety detector through instruction tuning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10421–10436.

Wei Zhao, Zhe Li, Yige Li, Ye Zhang, and Jun Sun. 2024. Defending large language models against jailbreak attacks via layer-specific editing. *arXiv preprint arXiv:2405.18166*.

Sicheng Zhu, Ruiyi Zhang, Bang An, Gang Wu, Joe Barrow, Zichao Wang, Furong Huang, Ani Nenkova, and Tong Sun. 2023. Autodan: interpretable gradient-based adversarial attacks on large language models. *arXiv preprint arXiv:2310.15140*.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J Zico Kolter, and Matt Fredrikson. 2023. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*.

A Related Works

A.1 Reducing Toxicity in LLMs

Existing approaches for reducing toxicity in large language models (LLMs) can be broadly categorized into three groups. (1) Pre-training Data Modification. These methods reduce toxic generation by curating or modifying the data used during model pre-training (Korbak et al., 2023; Keskar et al., 2019). (2) Tuning-based Methods. This line of work fine-tunes LLMs into safer variants using supervised learning or reinforcement learning from human feedback, such as Supervised Safety Fine-Tuning (SSFT) (Ouyang et al., 2022) and Direct Preference Optimization (DPO) (Rafailov et al., 2023). (3) Toxicity Detection and Filtering. These approaches add detection mechanisms to identify and block toxic content at the input or output level during inference (Zhang et al., 2023; Qin et al., 2020; Hallinan et al., 2022).

Above methods do not address the underlying causes of toxicity within the model, and aligned LLMs remain susceptible to adversarial prompting attacks (Zou et al., 2023; Zhu et al., 2023; Yan et al., 2025). Consequently, recent research has shifted toward analyzing the internal mechanisms of LLMs, with the goal of understanding and localizing the regions responsible for toxic behavior (Lee et al., 2024; Suau et al., 2024; Pan et al., 2025; Uppaal et al., 2025; Wang et al., 2024).

A.2 Mechanistic Interpretability

The goal of mechanistic interpretability is to reverse-engineer model behaviors (Elhage et al., 2021) by mapping functional properties, such as knowledge (Meng et al., 2022), linguistic features (Wei et al., 2024), toxicity (Wang et al., 2024), even tasks(Todd et al., 2023) to identifiable components within LLMs. These components include neurons (Yu and Ananiadou, 2023; Dai et al., 2022), Multi-headed Self-attention (MHSA) (Leong et al., 2023), Feed-Forward Network (FFN) (Deng et al., 2024; Duan et al., 2025), Transformer layer (Xu et al., 2024; Zhao et al., 2024), and circuit (Yao et al., 2024; Ou et al., 2025).

B Experimental Detail

In this section, we describe the implementation details for all baseline and proposed methods.

For **DPO**, we follow the setup of (Lee et al., 2024) and train models on 2,000 pairwise toxic samples. Default hyperparameters are used with $\beta = 0.1$. For larger models, we apply LoRA (Hu et al., 2021) to each layer, with a rank of 64, scaling factor of 16, and dropout rate of 0.1. Training employs early stopping with a patience value of 10 based on validation loss.

For **SSFT**, we follow the DPO setup, including dataset, LoRA, and early stopping.

For **ProFS**, we follow (Uppaal et al., 2025) and train on 500 pairwise toxic samples. Two hyperparameters are tuned: the number of top-k right singular vectors for constructing the toxic subspace, and the starting layer index ℓ_0 for projection-based editing. Specifically, we set $(k = 2,\ \ell_0 = 11)$

Table 5: GloSS Hyperparameters. τ and η are used to identify the global toxic subspace, while ℓ_0 determines the layers where projection is applied.

Model	τ	η	ℓ_0
GPT-2 Medium	1.0	0.8	13-24
GPT-J 6B	4.0	0.7	15-28
OPT-6.7B	2.0	0.8	10-32
Mistral-7B	1.0	0.7	15-32

for GPT-2; $(k = 10,\ \ell_0 = 11)$ for GPT-J; and $(k = 10,\ \ell_0 = 15)$ for all other models.

For **GloSS**, we introduce three hyperparameters: the toxicity threshold τ for selecting candidate directions, the variance ratio η for PCA-based subspace extraction, and the starting layer index ℓ_0 for applying projection. The detailed configurations of these hyperparameters for each model are summarized in Table 5.