REPO: Detoxifying LLMs via Representation Erasure-based Preference Optimization

Anonymous Authors[1]

Abstract

Large language models (LLMs) trained on web-scale data can produce toxic outputs, raising concerns for safe deployment. Prior defenses based on DPO, NPO, and similar algorithms reduce the likelihood of harmful continuations but not robustly: they are vulnerable to adversarial prompting and relearning attacks, and linear probing reveals that harmful "directions" remain present in representations. We propose Representation Erasure-based Preference Optimization (REPO), which reformulates detoxification as a token-level preference problem, forcing the representations of toxic continuations to converge toward their benign counterparts. Unlike baselines, REPO induces deep, localized edits to toxicity-encoding neurons while preserving utility, achieving state-of-the-art robustness against relearning attacks and enhanced GCG jailbreaks where existing methods fail.

1. Introduction

LLMs trained on massive, uncurated corpora can exhibit undesirable behaviors including toxic language generation (Wen et al., 2023), hazardous knowledge regurgitation (Li et al., 2024), and social bias amplification (Sheng et al., 2019; Gehman et al., 2020), motivating a growing set of alignment and detoxification algorithms. However, many such interventions are *fragile*: models remain vulnerable to jailbreak attacks such as GCG (Singh et al., 2025; Schwinn et al., 2024; Zou et al., 2023; Jia et al., 2024).

Unlearning has emerged as a complementary strategy (Liu et al., 2025; Xu et al., 2023) that aims to *remove hazardous capabilities* from models entirely, making them inaccessible even to adversaries with white- or black-box access. Early results suggest partial robustness; for example, RMU (Li et al., 2024) shows improved resistance to linear probing (Burns et al., 2023) and adversarial prompting (Zou et al., 2023; Huu-Tien et al., 2025). However, these methods fail against more adaptive jailbreaks (Łucki et al., 2025; Singh et al., 2025; Hu et al., 2024): *relearning* attacks recover supposedly removed capabilities through lightweight fine-tuning on as few as ten unrelated examples (Hu et al., 2024), and *enhanced* GCG variants substantially improve attack success with only small modifications (Łucki et al., 2025). These results suggest that reducing the likelihood of harmful outputs is often easier than removing the *internal representational affordances* that enable harmful generation.

Motivated by these vulnerabilities, recent work has explored representation-based approaches that intervene directly on hidden representations. Embedding-based unlearning is more resilient to paraphrasing attacks (Spohn et al., 2025), mechanistic localization improves robustness against relearning (Guo et al., 2025), and representation-level interventions resist membership inference and inversion attacks (Hu et al., 2025)—collectively suggesting that targeting hidden features enables more durable forgetting than output- or gradient-based approaches (Muhamed et al., 2025; Jung et al., 2025; Wang et al., 2025b).

A natural formalization of this idea is *representation erasure*: remove decodable information about an undesirable attribute from internal states so that downstream computation cannot reliably act on it. In classification, methods such as SURE (Sepahvand et al., 2025) provide evidence that adversarial invariance objectives can yield robust forgetting. Translating this to generative LLMs requires a fundamental shift: unlike classification—where one can "scrub" a single representation vector—detoxifying an autoregressive model requires controlling toxic features *at the token level* within a continuous generation stream.

This paper proposes *Representation Erasure-based Preference Optimization* (REPO), which adapts representation erasure to generative detoxification using *pairwise* supervision. For each prompt x_p, we assume a preferred *retain* continuation x_r (nontoxic) and a dispreferred *forget* continuation x_f (toxic). REPO combines two objectives: (i) a token-level anchoring loss that matches the edited model to a frozen

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

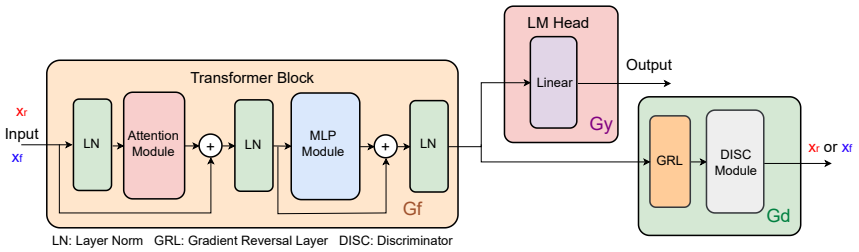

Figure 1. A schematic representation of REPO. Its regressor can be attached to any transformer block M targeted for unlearning; here, M is taken as the final transformer block before the linear unembedding layer. For each prompt, the retain (nontoxic) continuation x_r and the forget (toxic) continuation x_f are fed into the network, and the discriminator is trained to distinguish between toxic and nontoxic inputs.

reference model on retain continuations, preserving benign behavior, and (ii) a token-granular adversarial objective that makes retain and forget token representations indistinguishable, removing the features that distinguish harmful tokens. Like DPO (Lee et al., 2024), REPO leverages pairwise supervision, but enforces preferences in *representation space* rather than output space, removing the internal features that distinguish toxic sequences and rendering REPO robust against adaptive prompting and fine-tuning attacks.

Our contributions are as follows:

- We introduce REPO, a pairwise, token-level representation-erasure objective for detoxifying LLMs that couples reference anchoring on benign text with adversarial invariance between retain and forget representations.
- We evaluate REPO under adaptive recovery settings, including relearning and enhanced jailbreak attacks (Hu et al., 2024; Łucki et al., 2025), demonstrating superior detoxification and robustness compared to state-of-the-art, while preserving utility.

2. REPO for Detoxifying LLMs

We assume a paired dataset of triples

$$\mathcal{D} = \{(x_p^{(i)}, x_r^{(i)}, x_f^{(i)})\}_{i=1}^N,$$

where each prompt x_p is paired with a *retain* continuation x_r (non-toxic) and a *forget* continuation x_f (toxic). We write the corresponding full sequences as $s_r = [x_p; x_r]$ and $s_f = [x_p; x_f]$, then assign a domain label $d(s) \in \{0, 1\}$ with $d(s_r) = 0$ and $d(s_f) = 1$.

Our goal is to edit the model to (i) preserve the original model's behavior on retain data, and (ii) remove representational features that enable toxic generation on forget data.

2.1. Model components

Let the LLM be decomposed into (i) a transformer feature extractor $G_f(\cdot; \theta_f)$ that maps an input sequence s to token representations $\{h_t(s)\}_{t=1}^{|s|}$, and (ii) an LM head $G_y(\cdot; \theta_y)$ that maps $h_t(s)$ to logits $z_t(s)$ and next-token distributions

$$\pi_\theta(\cdot \mid s_{<t}) = \pi_\theta^t(s), \qquad \text{where } \theta = (\theta_f, \theta_y).$$

We also define a frozen reference model θ^{ref} (the original pretrained parameters), used only for anchoring.

To implement representation erasure, we attach a small discriminator (e.g., a two-layer MLP) $G_d(\cdot; \theta_d)$ to token representations at a chosen transformer layer ℓ (in our experiments, the final transformer block before unembedding). The discriminator is connected through a *gradient reversal layer* $R(\cdot)$ (Ganin et al., 2016), which is the identity on the forward pass and multiplies gradients by -1 on the backward pass. The discriminator outputs a domain probability

$$q_t(s) = G_d\big(R(h_t^{(\ell)}(s)); \theta_d\big) \in (0, 1),$$

interpreted as $q_t(s) \approx \Pr(d(s) = 1 \mid h_t^{(\ell)}(s))$.

Why representation erasure affects generation (informal rationale). In an autoregressive LM, the next-token distribution is a function of the hidden representation used by the LM head. In standard transformer LMs, logits are linear in the final representation, $z_t = W h_t + b$. If, for a fixed prompt, representations along toxic continuations are driven to match those along benign continuations at the layer feeding the head, then the logits—and therefore the next-token distributions—match as well, preventing the model from reliably continuing along a toxic trajectory beyond what the benign trajectory would produce. REPO approximates this matching by making retain and forget representations *indistinguishable* to a discriminator, while explicitly anchoring retain behavior to prevent trivial collapse.

2.2. REPO objective

Retain anchoring loss (token-level KL). To preserve benign behavior, we minimize a token-wise KL divergence between the edited model and the frozen reference model on retain sequences:

$$\mathcal{L}_{\text{retain}}(\theta) := \mathbb{E}_{s_r \sim \mathcal{D}_r}\left[\frac{1}{|s_r|}\sum_{t=1}^{|s_r|} \mathrm{KL}\Big(\pi_{\theta^{\text{ref}}}^t(s_r) \,\Big\|\, \pi_\theta^t(s_r)\Big)\right],$$

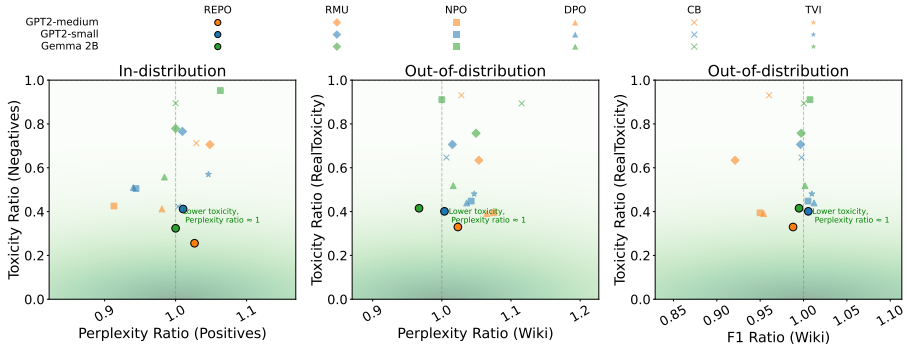

Figure 2. Detoxified models vs reference. **(Left)** Perplexity vs. toxicity ratios on PairToxicity (in-distribution); **(Middle)** Perplexity vs. toxicity ratios on WikiText/RealToxicity (OOD); **(Right)** F_1 ratio on WikiText vs. OOD toxicity. Each point is a model–method pair. The green gradient highlights lower toxicity and ratios near 1, darkest at the ideal point $(x=1,\ y=0)$. Dashed gray lines mark ratio = 1 for easy comparison to the reference.

where $\mathcal{D}_r$ denotes the retain distribution induced by $\mathcal{D}$. (The key property is token-level anchoring to the reference distribution across the entire prompt+retain sequence; either consistent KL direction may be used.)

Representation erasure loss (token-level domain adversarial). We train the discriminator to predict whether a token representation came from a retain or forget sequence, and simultaneously train the LLM to *fool* the discriminator. Using binary cross-entropy (BCE),

$$\mathcal{L}_{\text{dom}}(\theta_f, \theta_d) = \mathbb{E}_{s\sim\mathcal{D}_r\cup\mathcal{D}_f}\left[\frac{1}{|s|}\sum_{t=1}^{|s|}\text{BCE}\big(q_t(s),\ d(s)\big)\right],$$

where $\mathcal{D}_f$ denotes the forget distribution induced by $\mathcal{D}$.

Minimax form and implemented training loss. The intended optimization is

$$\begin{aligned}&\min_{\theta_f,\theta_y} \quad \alpha\,\mathcal{L}_{\text{retain}}(\theta) \;-\; (1-\alpha)\,\mathcal{L}_{\text{dom}}(\theta_f,\theta_d),\\ &\min_{\theta_d} \quad \mathcal{L}_{\text{dom}}(\theta_f,\theta_d),\end{aligned}$$

where $\alpha \in [0,1]$ trades off utility preservation and erasure pressure. In practice, we compute $\mathcal{L}_{\text{dom}}$ normally but connect G_d to G_f through $R(\cdot)$, which flips the gradient sign flowing into θ_f and implements the minimax without explicit alternating maximization steps (See Algorithm 1 for a formal summary of the method described above.)

3. Evaluation Metrics

We evaluate our approach along two complementary dimensions: (i) its effectiveness in removing toxic behaviors while preserving general capabilities; this is often referred to as unlearning-utility trade-off, and (ii) its robustness against adaptive attacks aimed at reactivating toxic behaviors. Below we describe the metrics used in each case.

3.1. Effectiveness

We measure effectiveness along two axes. **Toxicity Score**: we use the Perspective API (Geva et al., 2022; Lee et al., 2024), which estimates the probability a continuation would be perceived as toxic. **Utility**: we use Perplexity and F_1 score on WikiText-2 (Merity et al., 2017), where perplexity proxies divergence from the pretrained distribution and F_1 measures overlap with ground-truth continuations (see Section C.1).

3.2. Robustness

A key challenge in unlearning is robustness: toxic behavior may disappear, only for an adversary to recover it. We consider three attack strategies studied in the unlearning literature (Wang et al., 2025a; Łucki et al., 2025; Hu et al., 2024): relearning, orthogonalization, and enhanced GCG. For the latter two, model weights remain frozen and only inference-time manipulations are applied, whereas relearning modifies the model via fine-tuning.

Relearning Attack. Fine-tuning can easily reverse alignment or unlearning, even on small datasets with low mutual information with the forget set (Wang et al., 2025a; Łucki et al., 2025; Hu et al., 2024; Siddiqui et al., 2025). Following Łucki et al. (2025), we fine-tune unlearned models under two configurations: (i) on 10 forget-set examples (minimal direct exposure), and (ii) on 1000 retain-set examples (low mutual information with forgotten knowledge).

Orthogonalization Attack. Safety interventions can often be attributed to a direction in activation space (Arditi et al., 2024), an idea extended to unlearning by Łucki et al. (2025). Following their approach, we compute an *unlearned direction* for each transformer block as the difference in mean activations between the reference and unlearned models on the forget set (Belrose, 2023), then project this direction out of hidden representations at inference time, removing the

Table 1. Robustness of unlearning methods on GPT-2 (Medium) and Gemma-2B.Robustness of unlearning methods on GPT-2 (Medium) and Gemma-2B. Each cell reports post-attack toxicity (pre-attack in parentheses). Baseline toxicity before unlearning — GPT-2: 0.281/0.513; Gemma-2B: 0.208/0.486 (PairToxicity/RealToxicity).

		REPO	NPO	DPO	RMU	CB
GPT-2	Relearning Forget (PairToxicity)	**.169(.116)**	.202(.143)	.200(.144)	.253(.215)	.438(.120)
	Relearning Retain (PairToxicity)	**.119(.116)**	.148(.143)	.148(.144)	.204(.215)	.124(.120)
	Relearning Forget (RealToxicity)	**.294(.206)**	.377(.230)	.357(.224)	.463(.363)	.678(.332)
	Relearning Retain (RealToxicity)	**.207(.206)**	.245(.230)	.237(.224)	.362(.363)	.314(.332)
	Enhanced-GCG (RealToxicity)	**.208(.206)**	.347(.230)	.660(.224)	.389(.363)	.393(.332)
	Orthogonalization (RealToxicity)	**.308(.206)**	.335(.230)	.315(.224)	.525(.363)	.335(.332)
Gemma-2B	Relearning Forget (PairToxicity)	**.108(.083)**	.255(.247)	.169(.146)	.329(.206)	.161(.160)
	Relearning Retain (PairToxicity)	**.089(.083)**	.249(.247)	.169(.146)	.212(.206)	.162(.160)
	Relearning Forget (RealToxicity)	**.257(.215)**	.461(.439)	.304(.244)	.579(.356)	.402(.412)
	Relearning Retain (RealToxicity)	**.216(.215)**	.453(.439)	.304(.244)	.344(.356)	.421(.412)
	Enhanced-GCG (RealToxicity)	**.217(.215)**	.472(.439)	.269(.244)	.358(.356)	.428(.412)
	Orthogonalization (RealToxicity)	**.217(.215)**	.442(.439)	.248(.244)	.357(.356)	.415(.412)

offset introduced by unlearning.

Enhanced GCG Attack. Classic GCG attacks have been reported ineffective against representation-based unlearning methods such as RMU (Li et al., 2024; Łucki et al., 2025). We therefore adopt an enhanced variant that specifically targets unlearning defenses (Łucki et al., 2025). Rather than minimizing the standard loss toward a fixed affirmative target string (Zou et al., 2023), the attack leverages the reference model as a malicious teacher: adversarial prefixes are optimized with a distillation loss that aligns the unlearned model's hidden representations with those of the reference model (Thompson & Sklar, 2024), enabling recovery of harmful behaviors that classic GCG cannot elicit.

4. Experimental Details

Data and Models. Our evaluation relies on three datasets serving complementary purposes: a pairwise toxicity dataset for unlearning, PairToxicity, (Lee et al., 2024), WikiText-2 (Merity et al., 2017) for measuring generation quality, and RealToxicityPrompts (Gehman et al., 2020) for assessing OOD toxicity. We evaluate our approach on GPT-2 Small, GPT-2 Medium (Radford et al., 2019) and Gemma 2B (base) (Team et al., 2024). See Section C for further details.

Baselines. We compare REPO against three families of methods. *Steering-based*: Toxic Vector Intervention (TVI; Lee et al., 2024) subtracts identified toxic vectors from activations at inference time without retraining. *Output-space fine-tuning*: Direct Preference Optimization (DPO) and Negative Preference Optimization (NPO; Wang et al., 2025a; Łucki et al., 2025) shift output likelihoods toward preferred continuations. *Representation-space fine-tuning*: Representation Misdirection for Unlearning (RMU; Li et al., 2024; Huu-Tien et al., 2025; Kadhe et al., 2024) maps toxic directions to random ones, and Circuit Breakers (CB; Zou et al., 2024) severs causal pathways associated with harmful behavior; both are adapted here to detoxification.

5. Performance Evaluation

We evaluate REPO along two axes: the unlearning-utility trade-off (in-distribution on PairToxicity and OOD on RealToxicityPrompts and WikiText-2) and robustness to adversarial attacks. Additional results including qualitative results, mechanistic analysis, and ablations are provided in Sections D, F and G.

Mitigating Toxicity vs Preserving Utility Fig. 2 reports in-distribution results (PairToxicity). For GPT2-Small, REPO achieves the lowest toxicity on forget samples (0.096), substantially outperforming NPO (0.139), DPO (0.151), and RMU (0.153), while retain toxicity remains comparable. Perplexity results show that REPO increases uncertainty on toxic continuations (70.8 vs. 18.1 for the reference) while leaving retain perplexity largely unchanged, confirming targeted erasure without impairing general language modeling. For OOD evaluation, REPO yields the lowest RealToxicityPrompts score (0.21 vs. 0.24 for NPO) while virtually matching the reference model's utility (WikiText PPL 23.6), with both trends generalizing across GPT2-Medium and Gemma-2B.

Robustness to Attacks. Table 1 evaluates robustness under adversarial attacks for GPT2-Medium and Gemma-2B. REPO consistently outperforms all baselines across all attack types. For GPT2-Medium under relearning, REPO achieves lower toxicity on both retain (0.207 vs. 0.245 for NPO) and forget samples (0.119 vs. 0.148 for NPO and DPO). Against enhanced-GCG, REPO again achieves the lowest toxicity (0.208 vs. 0.389 for RMU and 0.347 for NPO), and the same trend holds for orthogonalization and Gemma-2B. In addition, further relearning attack results are in Section E.

5. Discussion

Current alignment techniques mask toxic capabilities without removing them. By reformulating detoxification as token-level representation erasure, REPO achieves durability against relearning and adaptive jailbreaks where output-based baselines fail, suggesting safety interventions require moving beyond behavioral preference optimization toward rigorous representation engineering.

References

Arditi, A., Obeso, O., Syed, A., Paleka, D., Panickssery, N., Gurnee, W., and Nanda, N. Refusal in language models is mediated by a single direction. *Advances in Neural Information Processing Systems*, 37:136037–136083, 2024.

Belrose, N. Diff-in-means concept editing is worst-case optimal: Explaining a result by Sam Marks and Max Tegmark. https://blog.eleuther.ai/diff-in-means/, 2023. Accessed: September 12, 2024.

Burns, C., Ye, H., Klein, D., and Steinhardt, J. Discovering latent knowledge in language models without supervision. In *Proc. Int. Conf. on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=ETKGuby0hcs.

Dathathri, S., Madotto, A., Lan, J., Hung, J., Frank, E., Molino, P., Yosinski, J., and Liu, R. Plug and play language models: A simple approach to controlled text generation. In *Int. Conf. on Learning Representations (ICLR)*, 2020.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35, 2016.

Gehman, S., Gururangan, S., Sap, M., Choi, Y., and Smith, N. A. RealToxicityPrompts: Evaluating neural toxic degeneration in language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 3356–3369, 2020.

Geva, M., Caciularu, A., Wang, K., and Goldberg, Y. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. In *Proceedings of the 2022 Annual Meeting of the Association for Computational Linguistics (ACL)*, 2022.

Guo, P., Syed, A., Sheshadri, A., Ewart, A., and Dziugaite, G. K. Mechanistic unlearning: Robust knowledge unlearning and editing via mechanistic localization. In *Proc. Int. Conf. Machine Learning (ICML)*, 2025.

Hu, J., Huang, Z., Yin, X., Ruan, W., Cheng, G., Dong, Y., and Huang, X. Falcon: Fine-grained activation manipulation by contrastive orthogonal unalignment for large language model, 2025.

Hu, S., Fu, Y., Wu, S., and Smith, V. Unlearning or obfuscating? jogging the memory of unlearned llms via benign relearning. In *Proc. of the Int. Conf. Learning Representations (ICLR)*, 2024.

Huu-Tien, D., Pham, T., Thanh-Tung, H., and Inoue, N. On effects of steering latent representation for large language model unlearning. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence and Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence and Fifteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'25/IAAI'25/EAAI'25. AAAI Press, 2025. ISBN 978-1-57735-897-8. doi: 10.1609/aaai.v39i22.34544. URL https://doi.org/10.1609/aaai.v39i22.34544.

Jia, X., Pang, T., Du, C., Huang, Y., Gu, J., Liu, Y., Cao, X., and Lin, M. Improved techniques for optimization-based jailbreaking on large language models, 2024.

Jung, J., Jung, B., Bae, S., and Lee, D. Opc: One-point-contraction unlearning toward deep feature forgetting, 2025.

Kadhe, S. R., Ahmed, F., Wei, D., Baracaldo, N., and Padhi, I. Split, unlearn, merge: Leveraging data attributes for more effective unlearning in llms, 2024.

Lee, A., Bai, X., Pres, I., Wattenberg, M., Kummerfeld, J. K., and Mihalcea, R. A mechanistic understanding of alignment algorithms: A case study on DPO and toxicity. In *Proc. Int. Conf. Machine Learning (ICML)*, 2024.

Li, N., Pan, A., Gopal, A., Yue, S., Berrios, D., Gatti, A., Li, J. D., Dombrowski, A.-K., Goel, S., Mukobi, G., Helm-Burger, N., Lababidi, R., Justen, L., Liu, A. B., Chen, M., Barrass, I., Zhang, O., Zhu, X., Tamirisa, R., Bharathi, B., Herbert-Voss, A., Breuer, C. B., Zou, A., Mazeika, M., Wang, Z., Oswal, P., Lin, W., Hunt, A. A., Tienken-Harder, J., Shih, K. Y., Talley, K., Guan, J., Steneker, I., Campbell, D., Jokubaitis, B., Basart, S., Fitz, S., Kumaraguru, P., Karmakar, K. K., Tupakula, U., Varadharajan, V., Shoshitaishvili, Y., Ba, J., Esvelt, K. M., Wang, A., and Hendrycks, D. The WMDP benchmark: Measuring and reducing malicious use with unlearning. In *Proc. Int. Conf. Machine Learning (ICML)*, 2024. URL https://openreview.net/forum?id=xlr6AUDuJz.

Liu, S., Yao, Y., Jia, J., Casper, S., Baracaldo, N., Hase, P., Yao, Y., Liu, C. Y., Xu, X., Li, H., Varshney, K. R., Bansal, M., Koyejo, S., and Liu, Y. Rethinking machine unlearning for large language models. *Nature Machine Intelligence*, 7(2):181–194, 2025. ISSN 2522-5839. doi: 10.1038/s42256-025-00985-0. URL https://doi.org/10.1038/s42256-025-00985-0.

Łucki, J., Wei, B., Huang, Y., Henderson, P., Tramèr, F., and Rando, J. An adversarial perspective on machine unlearning for AI safety. *Transactions on Machine Learning Research*, 2025. ISSN 2835–8856. URL https://openreview.net/forum?id=J5IRyTKZ9s.

Merity, S., Xiong, C., Bradbury, J., and Socher, R. Pointer sentinel mixture models. In *Proc. of the Int. Conf. on Learning Representations (ICLR)*, 2017.

Muhamed, A., Bonato, J., Diab, M. T., and Smith, V. SAEs can improve unlearning: Dynamic sparse autoencoder guardrails for precision unlearning in LLMs. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=kaPAalWAp3.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Schwinn, L., Dobre, D., Xhonneux, S., Gidel, G., and Gunnemann, S. Soft prompt threats: Attacking safety alignment and unlearning in open-source llms through the embedding space. In *Advances in Neural Information Processing Systems (38)*, 2024. URL https://openreview.net/forum?id=CLxcLPfARc.

Sepahvand, N. M., Dziugaite, G. K., Triantafillou, E., and Precup, D. Selective unlearning via representation erasure. In *Proc. of the Int. Conf. Learning Representations (ICLR)*, 2025.

Sheng, E., Chang, K.-W., Natarajan, P., and Peng, N. The woman worked as a babysitter: On biases in language generation. In Inui, K., Jiang, J., Ng, V., and Wan, X. (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3407–3412, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1339. URL https://aclanthology.org/D19-1339/.

Siddiqui, S. A., Weller, A., Krueger, D., Dziugaite, G. K., Mozer, M. C., and Triantafillou, E. From dormant to deleted: Tamper-resistant unlearning through weight-space regularization. 2025.

Singh, N. D., Müller, M., Croce, F., and Hein, M. Unlearning that lasts: Utility-preserving, robust, and almost irreversible forgetting in llms, 2025.

Spohn, P., Girrbach, L., Bader, J., and Akata, Z. Align-then-unlearn: Embedding alignment for llm unlearning, 2025.

Team, G., Riviere, M., Pathak, S., Sessa, P. G., Hardin, C., Bhupatiraju, S., Hussenot, L., Mesnard, T., Shahriari, B., Ramé, A., et al. Gemma 2: Improving open language models at a practical size, 2024.

Thompson, T. B. and Sklar, M. Flrt: Fluent student-teacher redteaming, 2024.

Wang, C., Zhang, Y., Jia, J., Ram, P., Wei, D., Yao, Y., Pal, S., Baracaldo, N., and Liu, S. Invariance makes LLM unlearning resilient even to unanticipated downstream fine-tuning, 2025a.

Wang, X., Li, Z., Wang, B., Hu, Y., and Zou, D. Model unlearning via sparse autoencoder subspace guided projections, 2025b. URL https://arxiv.org/abs/2505.24428.

Wen, J., Ke, P., Sun, H., Zhang, Z., Li, C., Bai, J., and Huang, M. Unveiling the implicit toxicity in large language models. In Bouamor, H., Pino, J., and Bali, K. (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 1322–1338, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.84. URL https://aclanthology.org/2023.emnlp-main.84/.

Xu, H., Zhu, T., Zhang, L., Zhou, W., and Yu, P. S. Machine unlearning: A survey. *ACM Comput. Surv.*, 56(1), August 2023. ISSN 0360-0300. doi: 10.1145/3603620. URL https://doi.org/10.1145/3603620.

Zou, A., Wang, Z., Carlini, N., Nasr, M., Kolter, J. Z., and Fredrikson, M. Universal and transferable adversarial attacks on aligned language models, 2023.

Zou, A., Phan, L., Wang, J., Duenas, D., Lin, M., Andriushchenko, M., Wang, R., Kolter, Z., Fredrikson, M., and Hendrycks, D. Improving alignment and robustness with circuit breakers. In *Advances in Neural Information Processing Systems (38)*, 2024.

Algorithm 1 REPO training (one optimization step)

Require: Minibatch $\{(x_p^{(i)}, x_r^{(i)}, x_f^{(i)})\}_{i=1}^{B}$, reference model θ^{ref} (frozen), tradeoff $\alpha \in [0, 1]$
Require: Learning rates η (LLM) and η_d (discriminator)
Ensure: Updated LLM parameters $\theta = (\theta_f, \theta_y)$ and discriminator parameters θ_d
Form $s_r^{(i)} \leftarrow [x_p^{(i)}; x_r^{(i)}]$ and $s_f^{(i)} \leftarrow [x_p^{(i)}; x_f^{(i)}]$
Compute $\mathcal{L}_{\text{retain}}(\theta)$ on $\{s_r^{(i)}\}$ (token-wise KL to θ^{ref})
Compute $\mathcal{L}_{\text{dom}}(\theta_f, \theta_d)$ on $\{s_r^{(i)}, s_f^{(i)}\}$ (token-wise BCE with labels $d(s_r) = 0$, $d(s_f) = 1$)
$\theta_d \leftarrow \theta_d - \eta_d \nabla_{\theta_d} \mathcal{L}_{\text{dom}}$ {train discriminator}
$\theta \leftarrow \theta - \eta \nabla_{\theta} (\alpha\, \mathcal{L}_{\text{retain}} + (1 - \alpha)\, \mathcal{L}_{\text{dom}})$
    {GRL on discriminator input flips the sign of gradients into θ_f}

A. Questions We Anticipate

1. **Why Token-level granularity?** A sequence-level discriminator (e.g., pooling representations before classification) can encourage coarse alignment while allowing toxic information to remain localized in specific positions. REPO instead applies the discriminator *per token* and averages over tokens. This targets the parts of the computation graph that encode toxic tokens and their immediate causal footprint, while the retain KL discourages broad degradation of language modeling behavior.

2. **Why "preference optimization", and how REPO differs from DPO/NPO?** We call REPO a preference optimization method because the supervision is pairwise: for each prompt, we are given a preferred continuation x_r and a dispreferred continuation x_f. DPO/NPO use this pairing to shift *output-space* likelihoods, typically by increasing the relative log-probability of x_r over x_f (often with an implicit or explicit regularization toward a reference model). REPO uses the same pairing differently: it (i) anchors the model to the reference on preferred text via token-wise KL, and (ii) uses the rejected text to drive *representation-level* erasure through a domain-adversarial objective. This is designed to address recoverability: rather than only making toxic outputs less likely under current decoding, REPO removes decodable internal features that distinguish toxic continuations from benign ones under the same prompt. From that perspective, REPO can be viewed as an unlearning algorithm, unlike prior preference optimization methods that do not erase the knowledge of the dispreferred continuations from internal features.

3. **Why did you choose models like GPT-2 and Gemma-2B base for evaluation?** Our choice was deliberate: these models are lightweight enough to support detailed *layer- and token-level mechanistic analysis*, which is central to the paper's contribution. Importantly, REPO is *model-agnostic* and scales naturally: the method only requires access to intermediate representations and a discriminator. Our experiments offer a compelling proof-of-concept with deep mechanistic evidence. Importantly, REPO's behavior is consistent across two distinct architectures (GPT-2, Gemma), suggesting architectural generality. Many unlearning methods (e.g., RMU, CB) were first validated on smaller scales before scaling up; we view our work as establishing the mechanistic foundation for future large-scale extensions.

4. **Why did you not test REPO on larger instruction-tuned models like Llama-2-7B or Mixtral?** Our experiments deliberately focus on smaller open models (GPT-2, Gemma-2B) to allow exhaustive mechanistic analysis (layer–token drift, neuron activation shifts, weight-space distances). These analyses would have not been feasible on 13B+ models due to cost and reproducibility barriers. Our goal is to provide a controlled, mechanistic demonstration. Scaling REPO is conceptually straightforward: it requires only a discriminator on hidden states. We are releasing code so the community can apply it to larger aligned models.

5. **Why are there no human evaluations or alternative detectors for toxicity?** We agree that multiple evaluators would enrich the results. For this submission, we prioritized comparability with prior ICLR/NeurIPS papers by using Perspective API, ensuring our baselines are on equal footing. Crucially, REPO does not optimize against Perspective, so it is detector-agnostic. Our mechanistic evidence (localized neuron edits, deeper layer shifts) shows that REPO changes the model itself, not just a metric. We view this as a stronger and more general guarantee than detector-specific scores.

6. **Why are ablations focused on token- vs segment-level?** We prioritized the ablation most central to REPO's novelty: token-level discrimination. Other knobs (loss weighting, discriminator depth) have standard effects and do not alter the mechanistic story. Our weight- and neuron-level analyses already show that REPO's behavior differs qualitatively from prior methods, and these structural differences (not hyperparameter sweeps) are what account for its robustness. Further ablations are left to future work due to space constraints.

Table 2. Hyper-parameters used for each method and model. A dash (–) indicates the parameter is not applicable. For parameters listed as arrays in the configuration (e.g., two runs with 5×10^{-6} for NPO on Gemma-2B), the table specifies this explicitly.

Model	Method	Learning Rate (lr)	α	β	c
GPT-2-Small	REPO	2×10^{-6}	0.2	–	–
	DPO	1×10^{-6}	–	0.5	–
	NPO	1×10^{-6}	0.2	0.5	–
	RMU	5×10^{-6}	0.95	–	500
	CB	1×10^{-5}	100.0	–	–
GPT-2 Medium	REPO	5×10^{-6}	0.2	–	–
	DPO	1×10^{-6}	–	0.5	–
	NPO	1×10^{-6}	0.4	0.5	–
	RMU	5×10^{-6}	0.95	–	500
	CB	5×10^{-5}	100.0	–	–
Gemma-2B	REPO	5×10^{-5}	0.5	–	–
	DPO	1×10^{-5}	–	0.2	–
	NPO	5×10^{-6}	0.8	0.5	–
	RMU	5×10^{-5}	0.95	–	500
	CB	1×10^{-5}	1000.0	–	–

7. **Does the use of synthetic toxic/non-toxic pairs introduce bias or limit generalization?** Synthetic pairs (via PPLM and greedy decoding) allow us to control for semantic similarity while isolating toxicity, which is essential for training a representation-level discriminator. This setup minimizes confounds such as topic or length, ensuring that REPO learns to erase toxic features rather than spurious correlations. Importantly, REPO's robustness evaluations (orthogonalization, relearning, GCG jailbreaks) demonstrate generalization to settings far outside the synthetic training distribution. In addition, REPO achieves strong performance on naturally occurring toxic continuations (RealToxicityPrompts), indicating that it transfers beyond synthetic contrasts.

8. **Are the baseline comparisons (to DPO, NPO, CB, and RMU) fair, and why not include RLHF-tuned models?** We implemented DPO and NPO using standard hyperparameters from their original papers, verifying that our implementations match reported performance. For representation-level baselines (e.g., CB, RMU), we reproduced them faithfully to ensure apples-to-apples comparison. We did not include RLHF-tuned models because REPO is not intended as a competitor to RLHF; rather, it is complementary. RLHF requires extensive preference data and large-scale tuning, while REPO can be applied post-hoc as a lightweight safety repair that directly edits hidden states. Thus, our focus is on representation-level methods, which are the most natural comparators—but REPO can also be layered on top of RLHF-trained systems.

9. **Is the enhanced GCG attack too unrealistic as a threat model?** We agree that access to the reference model is not always realistic, but we deliberately stress-tested REPO under *worst-case white-box assumptions*. The fact that REPO resists these extreme attacks strengthens confidence in its robustness to weaker, more realistic black-box jailbreaks. Our framing follows the *cryptographic principle of testing against the strongest adversary available*.

10. **Where can I find hyperparameters and training details?** See Section B.

11. **Why do the experiments focus only on toxicity, rather than other unlearning tasks?** We chose toxicity as a *representative and socially urgent case study*. The method, however, is general: REPO only requires a binary discriminator on hidden states. In principle, it can be applied to any capability removal (e.g., memorized data, unsafe skills). We see our toxicity experiments as a *first demonstration*, with generalization left for follow-up work.

B. Reproducibility Statement

To facilitate reproducibility, we provide in Table 2 the exact hyper-parameters used for each method and model evaluated in this paper, together with their definitions. We also detail in Table 3 the training settings used across models, including the number of unlearning or relearning epochs, batch sizes, weight decay values, and other implementation choices. In

Table 3. Training settings and implementation details for unlearning and relearning experiments.

Model	Setting	Value	Notes
GPT-2 Small	Unlearning epochs	10	for all methods
	Batch size	128	for unlearning
	Weight decay	0.001	for all methods
	Relearning attack	wd = 1×10^{-5}, lr = 1×10^{-5}	
	Gradient clipping	max_norm = 10.0	DPO & NPO
GPT-2 Medium	Unlearning epochs	10	for all methods
	Batch size	64	for unlearning
	Weight decay	0.01	for all methods
	Relearning attack	wd = 1×10^{-5}, lr = 1×10^{-5}	
	Gradient clipping	max_norm = 10.0	DPO & NPO
Gemma-2B	Unlearning epochs	5	for all methods
	Batch size	16	for unlearning
	Weight decay	0.01	for all methods
	Relearning attack	wd = 1×10^{-4}, lr = 5×10^{-5}	
	Gradient clipping	max_norm = 10.0	DPO & NPO

addition, we describe the setup of our relearning attack experiments and the sampling procedures used for forget and retain sets. The full training and evaluation code will be released upon acceptance of the paper to enable independent verification and extension of our results.

Hyper-parameter definitions. Below we explain the roles of the hyper-parameters as used in our implementations (consistent with the original formulations when applicable):

- **lr:** learning rate used for parameter updates by the optimizer.
- **REPO — α:** weight on the adversarial (discriminator) loss relative to the KL/reference-matching loss. It controls the trade-off between preserving similarity to the reference model and aligning the forget representations toward the retain representations in the shared space.
- **DPO — β:** scaling factor applied to the difference in log probabilities between the model and reference ($\Delta \log p$); it sharpens or flattens the preference logit before the log-sigmoid. Higher β yields more aggressive preference gradients.
- **NPO — β:** scaling factor in the negative-preference term; α weights the forget loss relative to the standard LM loss on retain examples. Together they govern how strongly the model is pushed to forget and how much it is anchored to the retain examples.
- **RMU — α:** interpolation weight between forgetting and retaining representations. The hyper-parameter c defines the norm of the random "control" vector used to specify the forgetting direction against which the representation is aligned.
- **CB — α:** coefficient on the circuit-breaker loss relative to the retain loss, determining how strongly the model is penalized when inner-product activations associated with forget features deviate from the desired retain alignment.

Training and implementation details. Beyond the hyper-parameters in Table 2, Table 3 summarises the key training settings we used across models and methods. These include the number of unlearning epochs, batch sizes, weight decay values, and learning rates used for the "relearning attack" experiments. All unlearning runs used a linear learning-rate warm-up of 100 steps. For DPO and NPO, we additionally clamped the logits to a fixed range (-30 to +30) to prevent numerical overflow and applied gradient-norm clipping to improve training stability.

Relearning attack. For the relearning attack experiments, we fine-tuned the models for three epochs. We conducted two separate attack variants: (i) relearning on forget samples and (ii) relearning on retain samples. For the forget-based attack,

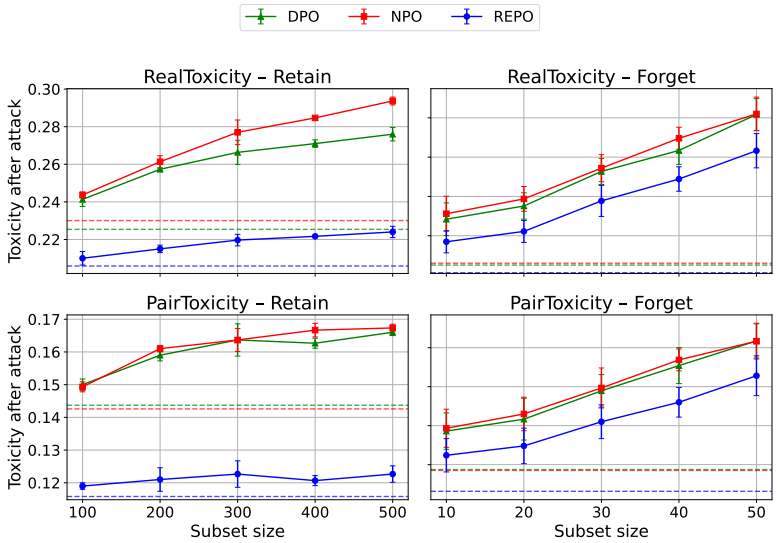

Figure 3. Average toxicity after the *Relearning Attack* for different subset sizes across methods on GPT2-small. **(Top)** OOD toxicity (RealToxicity); **(Bottom)** In-distribution toxicity (pairwise set). Dashed horizontal lines indicate each method's baseline toxicity before the attack.

we report the average over three independent runs, each using 10 randomly selected samples from the ToxicityPair dataset. For the retain-based attack, we likewise report the average over three runs using 100 randomly selected retain samples from the same dataset. In Fig. 3 we show trends as we vary the set sizes; specifically, forget sizes $\{10, 20, 30, 40, 50\}$ and retain sizes $\{100, 200, 300, 400, 500\}$. All reported values are averages over three independent runs.

C. Experimental Details

C.1. Utility Metrics

Utility is evaluated using perplexity and F_1 score on WikiText-2 (Merity et al., 2017) , a neutral dataset excluded from unlearning. Perplexity, defined as the exponentiated average negative log-likelihood of the ground-truth continuation, measures how well a model predicts reference text. We report perplexity for both the unlearned and the reference model, i.e. the original model before unlearning, which is regarded as a high-utility reference point; differences between them provide a proxy for divergence from the distribution of the original pretrained model. F_1 is defined as the harmonic mean of precision and recall, where precision is the fraction of generated tokens appearing in the ground-truth continuation, and recall is the fraction appearing in the model's generation.

C.2. Models and Data

Data. The pairwise dataset, introduced in Lee et al. (2024), contains 24,576 prompt–continuation pairs constructed from sentences in Wikitext. For each prompt, we generate two continuations: a nontoxic continuation via greedy decoding, which forms the *retain set*, and a toxic continuation using PPLM (Dathathri et al., 2020) guided by a toxicity probe, which forms the *forget set*. This construction yields a *pairwise dataset* in which every prompt is associated with both a toxic and a nontoxic continuation, providing aligned examples for unlearning.

To measure preservation of generation capabilities, we use Wikitext, a standard language modeling benchmark consisting of Wikipedia articles, for evaluating perplexity and F_1. To measure toxicity reduction, we use the RealToxicityPrompts challenge set, which contains 1,199 prompts designed to elicit toxic outputs from language models.

Models. GPT-2 Medium is an autoregressive transformer trained on OpenAI's WebText corpus without any subsequent alignment or safety tuning. For Gemma 2B, we use the publicly available base checkpoints, which are pretrained models not fine-tuned for instruction following or safety; the aligned variants of these families (e.g., Gemma-Instruct) are deliberately excluded to ensure that detoxification is evaluated from raw pretrained models. For optimization, we apply full-parameter finetuning to GPT-2 (Small and Medium) given their smaller sizes, while for Gemma 2B we employ parameter-efficient

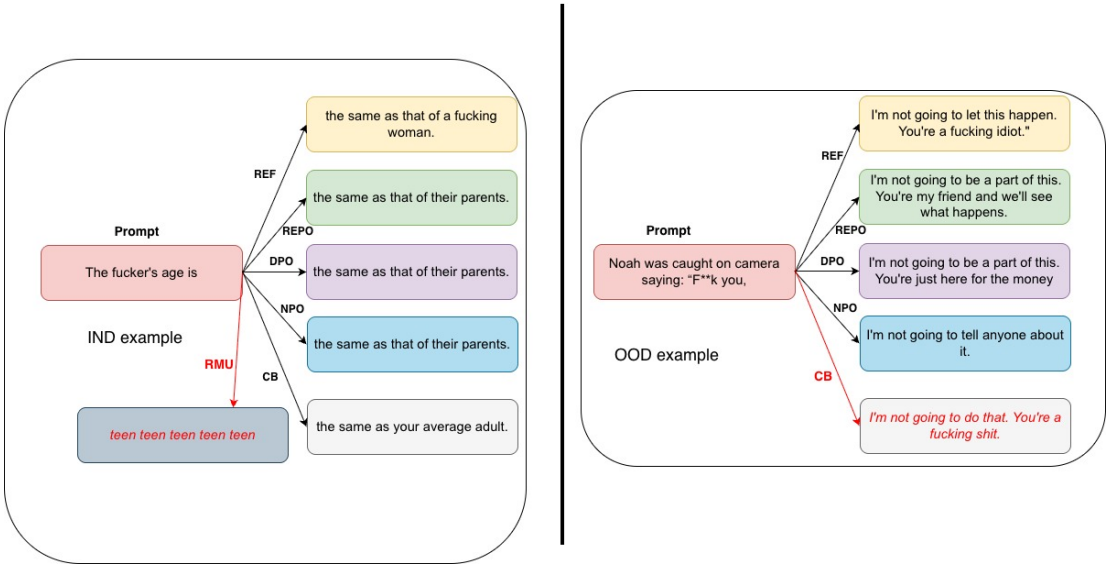

Figure 4. Generations from different unlearned models for a randomly selected prompt from the in-distribution dataset (PairToxicity, left) and the OOD dataset (RealToxicity, right). This illustrates how each model responds to toxic prompts after unlearning.

LoRA finetuning.

D. Qualitative Results

Fig 4 shows the generations of different unlearned models for a randomly selected prompt from the in-distribution dataset (PairToxicity, left) and a randomly selected prompt from the OOD dataset (RealToxicity, right). Visual inspection reveals two important patterns. First, RMU produces largely unintelligible outputs on toxic prompts, consistent with its extremely high perplexity on negative generations (2079.71 vs. 18.172 for the reference model). This indicates that RMU's intervention disrupts the model, producing gibberish instead of selectively removing toxic content. Second, CB reduces toxicity for in-distribution prompts (0.2814 vs. 0.4995 for the original model) but fails to generalize, with OOD toxicity remaining essentially unchanged (0.4995 vs. 0.5121). These results highlight that some unlearning methods either compromise fluency or lack robustness to OOD scenarios.

E. Relearning Results

For the relearning attack on GPT2-Small, Fig. 3 shows that REPO maintains a consistent advantage over DPO and NPO across different numbers of forget and retain samples. Together these results show that REPO resists the recovery of toxic behaviors even under stronger adversarial settings. s

F. Mechanistic Analyses

F.1. Changes in the Weight Space

We examine the magnitude of modifications each unlearning method imparts on the model's parameters. Fig. 5 plots the average relative L2 distance between the weights of the unlearned and reference models at each Transformer block. A clear pattern emerges: REPO induces substantially larger weight-space edits compared to both DPO and NPO. While all methods tend to modify later layers more than earlier ones, REPO's updates are significantly greater, particularly from the middle to the final blocks of the network. Siddiqui et al. (2025) recently showed that unlearning algorithms that yield a larger L2 distance from the original model exhibit increased robustness to relearning attacks, which is consistent with our observation that REPO is significantly more robust against those attacks compared to DPO and NPO. For REPO, the larger weight-space edits are due to the method's design, which applies adversarial pressure directly to the hidden representations of the final transformer block. This architectural choice concentrates the learning signal in the deeper layers, compelling more significant parametric adjustments to align toxic and non-toxic representations. In contrast, DPO and NPO, which

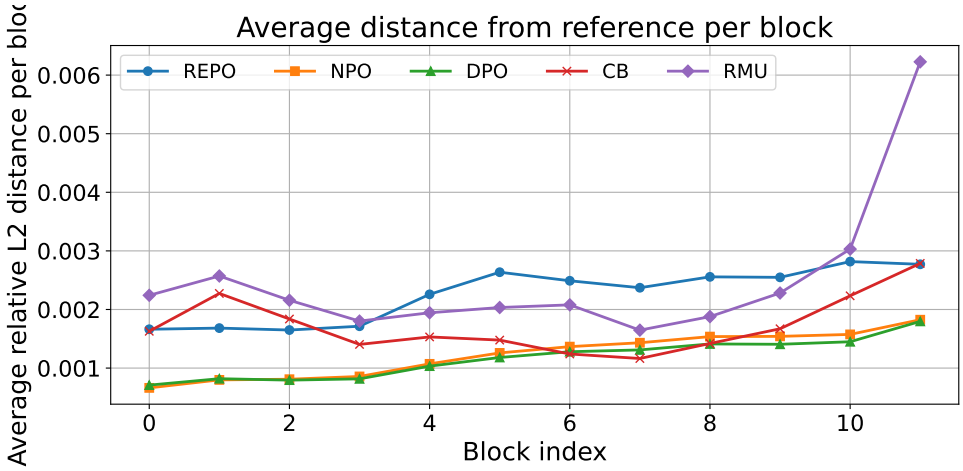

Figure 5. Average relative ℓ_2 distance between unlearned models and the reference model at each Transformer block for REPO, NPO, and DPO.

operate on output probabilities, distribute their updates more diffusely. While seemingly more disruptive, we will show in the following section that these larger weight-space modifications enable more precise, localized changes in the model's internal representations.

F.2. Changes in Key and Value vectors

Plots in Fig. 6 illustrate how each method affects the value and key vectors of the model across the top 2 000 neurons most aligned with the toxic vector W_{TOXIC}. Across all three methods (SURE, DPO, and NPO), the changes in both the value and key vectors are minimal, with cosine similarities between the pre- and post-unlearning weights remaining very close to one. For the most toxic neurons, our method induces a slightly larger reduction in cosine similarity, but this difference remains very subtle compared to the other two methods.

Despite the very subtle differences in key and value weight changes between our method and DPO/NPO, these small adjustments produce a markedly larger shift in the corresponding activations. Specifically, SURE yields a greater change in the key activations of those same neurons compared to DPO and NPO. In other words, even minor adjustments to the key and value weights, when guided by our adversarial alignment objective, are sufficient to shift the internal representations so that activations associated with toxic features are suppressed. This effect can be seen most clearly in the bottom row of Fig. 6, where the mean absolute activation difference increases sharply for neurons most strongly aligned with toxicity. This demonstrates that SURE achieves detoxification primarily through targeted changes in the internal activations, rather than large weight updates, resulting in a more precise and controlled unlearning effect.

F.3. The Effects on Representations and Weights

The analysis in this section is focused on studying the mechanisms behind REPO's performance. We demonstrate that REPO has larger magnitude weight edits (Section F.1), but these edits result in more localized edits on conditional distributions of toxic words, and affect representations deeper in the network. Building on the analysis by Lee et al. (2024), we then inspect the changes in value and key vectors, observing that the biggest shift happens in dimensions most *and least* aligned with toxic directions. Our ablations reveal that these differences between REPO and other methods are due to two key algorithm design choices: (1) edits on the representations instead of output, resulting in bigger changes deeper in the network, and (2) REPO's optimization objective being at a token-level granularity – this ensures more localized shifts on the toxic word distribution.

F.4. Changes in the intermediate states.

In Section F.1 we show that REPO makes larger edits in weight space. Having observed that, we now examine how these changes affect the model's intermediate representations. Fig. 8 visualizes this by plotting the representational drift (1-cosine similarity) between the unlearned and reference models' hidden states across all layers for a sample toxic continuation. The

heatmaps for REPO show that modifications are highly localized. Significant drift is concentrated in the network's deeper layers, and is confined almost exclusively to the columns corresponding to the toxic tokens, while the representations for adjacent tokens show minimal change. In stark contrast, DPO and NPO induce more diffuse, lower-magnitude changes that are spread across a broader set of tokens and layers. This analysis provides an intuition for REPO's good utility-unlearning trade-off: it achieves effective unlearning by making targeted modifications to the representations of specific toxic inputs while preserving the integrity of non-toxic ones.

G. Ablations of Algorithmic Components

We conduct a series of ablations to dissect REPO's design and identify the sources of its effectiveness: representation-space edits, and token-level objective. We then provide evidence that REPO more aggressively targets the specific neurons most aligned with toxicity compared to baselines.

Changing the token-level objective. To isolate REPO's components responsible for the localized edits, we conduct an ablation study on the granularity of REPO's adversarial objective. We compare our standard approach, where the discriminator evaluates each token's representation individually, with a variant where representations are averaged over non-overlapping segments before being passed to the discriminator. The results are visualized in Fig. 10. The top row, showing the standard token-level objective, exhibits the highly localized representational drift previously discussed. In contrast, the bottom row shows that using averaged segments causes this localization to vanish. The representational drift becomes diffuse, spreading across multiple tokens rather than being confined to specific ones. This diffusion in representation space correlates with a degradation in unlearning performance, yielding a worse utility-unlearning trade-off. This ablation provides strong evidence that the token-level granularity of REPO's adversarial loss is a key mechanism responsible for the precision of its edits, which in turn contributes to its strong performance.

The role of representation-based objective. Our analysis has shown that REPO's interventions are concentrated in deeper layers compared to output-space methods like DPO and NPO. To determine if this is a general property of representation-based unlearning, we now compare REPO with two other representation-based methods: Circuit Breakers (CB) and Representation Misdirection (RMU). The heatmaps in Fig. 8 (bottom row) confirm that this is indeed the case. All three representation-based methods predominantly alter the model in its later layers, suggesting that the depth of modification is a feature of targeting internal representations directly. However, the figure also reveals a critical distinction in the precision of these deep edits. While REPO's changes are localized to specific toxic tokens, the interventions from CB and RMU are not. CB's edits appear to impact entire layers indiscriminately, and RMU's are scattered broadly across both tokens and layers.

This comparison yields a key insight: while targeting representations focuses unlearning on deeper parts of the network, REPO's token-level adversarial objective is responsible for the localization necessary for effective detoxification, a property that these other representation-based methods lack.

Changes in neuron activations. Finally, we examine how each method alters neuron activations based on their semantic roles. Following prior work, we first identify a toxic direction, W_{toxic}, using linear probing on the reference model's representations. Fig. 7 shows the activation drift ($1 - \cos$ similarity) between the unlearned and reference models for a single negative prompt, comparing the top-10 most toxic neurons to randomly selected neurons, with changes plotted across tokens. Major changes occur primarily in the toxic neurons and for toxic tokens, while random neurons and non-toxic tokens remain largely unchanged. Fig. 9 extends this analysis across all negative prompts, showing the mean absolute activation change as a function of each neuron's alignment with W_{toxic}. This reveals a U-shaped pattern for all methods: the largest changes occur in neurons most aligned or anti-aligned with W_{toxic}, while neutrally aligned neurons are minimally affected. Crucially, REPO induces substantially larger changes in the neurons most aligned with the toxic direction compared to DPO or NPO, highlighting its ability to target toxic tokens and the neurons responsible for encoding toxic concepts.

Connection to prior work. We compare REPO to prior domain-adversarial methods: DANN (Ganin et al., 2016) and SURE (Sepahvand et al., 2025). DANN trains a domain regressor to distinguish source and target domains, while the feature extractor is adversarially updated to produce features that are task-discriminative but domain-invariant. SURE adapts this for selective unlearning in image classification, using a held-out set to erase representations that differentiate forget samples.

To evaluate REPO, we compare it to (1) a SURE-style objective on an identical model, (2) a baseline using only the token-wise cross-entropy (CE) loss on retain samples (to ensure improvements are not solely due to the retain loss), and (3)

Table 4. Comparison of GPT2-small unlearned with different objectives. Nontoxic / Toxic / RealToxicity: toxicity on in-distribution retain, in-distribution forget, and OOD RealToxicity datasets, respectively. PPL / F_1: perplexity and F_1 on WikiText.

Method	Nontoxic	Toxic	RealToxicity	PPL	F1
REF	0.0460	0.2824	0.5123	28.0379	0.1930
CE	0.0437	0.2367	0.4454	45.7689	0.1859
SURE	0.0418	0.2021	0.3750	34.0385	0.1869
REPO	0.0446	0.1020	0.1913	28.2314	0.1930

Table 5. Ablation on GPT2-small comparing linear and nonlinear discriminators in the domain-adversarial objective. Nontoxic, Toxic, and RealToxicity report toxicity on in-distribution retain samples, in-distribution forget samples, and OOD RealToxicity, respectively. Perplexity (PPL) and F_1 are evaluated on WikiText.

Method	Nontoxic	Toxic	RealToxicity	PPL	F1
Reference	0.0461	0.2812	0.5120	28.0379	0.1930
One-layer Disc.	0.0451	0.1356	0.2396	28.2567	0.1952
Two-layer Disc.	0.0446	0.1020	0.1913	28.2314	0.1937

the reference model (REF) prior to any unlearning. Results on GPT2-small are shown in Table 4. Using only the CE retain loss yields minor toxicity reduction but severely harms perplexity (28 $\rightarrow$ 34). The SURE objective improves toxicity further but underperforms REPO in perplexity. REPO, by contrast, achieves substantially stronger toxicity reduction (0.5123 $\rightarrow$ 0.1913) while matching the reference model's perplexity, demonstrating the effectiveness of combining the token-level KL retain loss with the adversarial forget objective.

Effect of discriminator capacity. We examine the impact of discriminator complexity in the domain-adversarial objective, comparing REPO's two-layer MLP to a linear variant on GPT2-small with all else fixed. Table 5 shows that both reduce toxicity relative to the reference model, but the two-layer discriminator achieves stronger reduction on in-distribution toxic samples and OOD prompts, while perplexity and F_1 on WikiText remain comparable. Toxicity on non-toxic (retain) samples is largely unchanged, suggesting that modest nonlinearity improves representation erasure without harming language modeling.

Importantly, this improvement comes at minimal computational cost. The two-layer discriminator is a small fully connected network with dimensions $D_{\text{model}} \rightarrow 16 \rightarrow 2$. Compared to the forward and backward passes of the language model, the additional computation is effectively zero, and overall training time is dominated by gradient computation in the LLM rather than the discriminator.

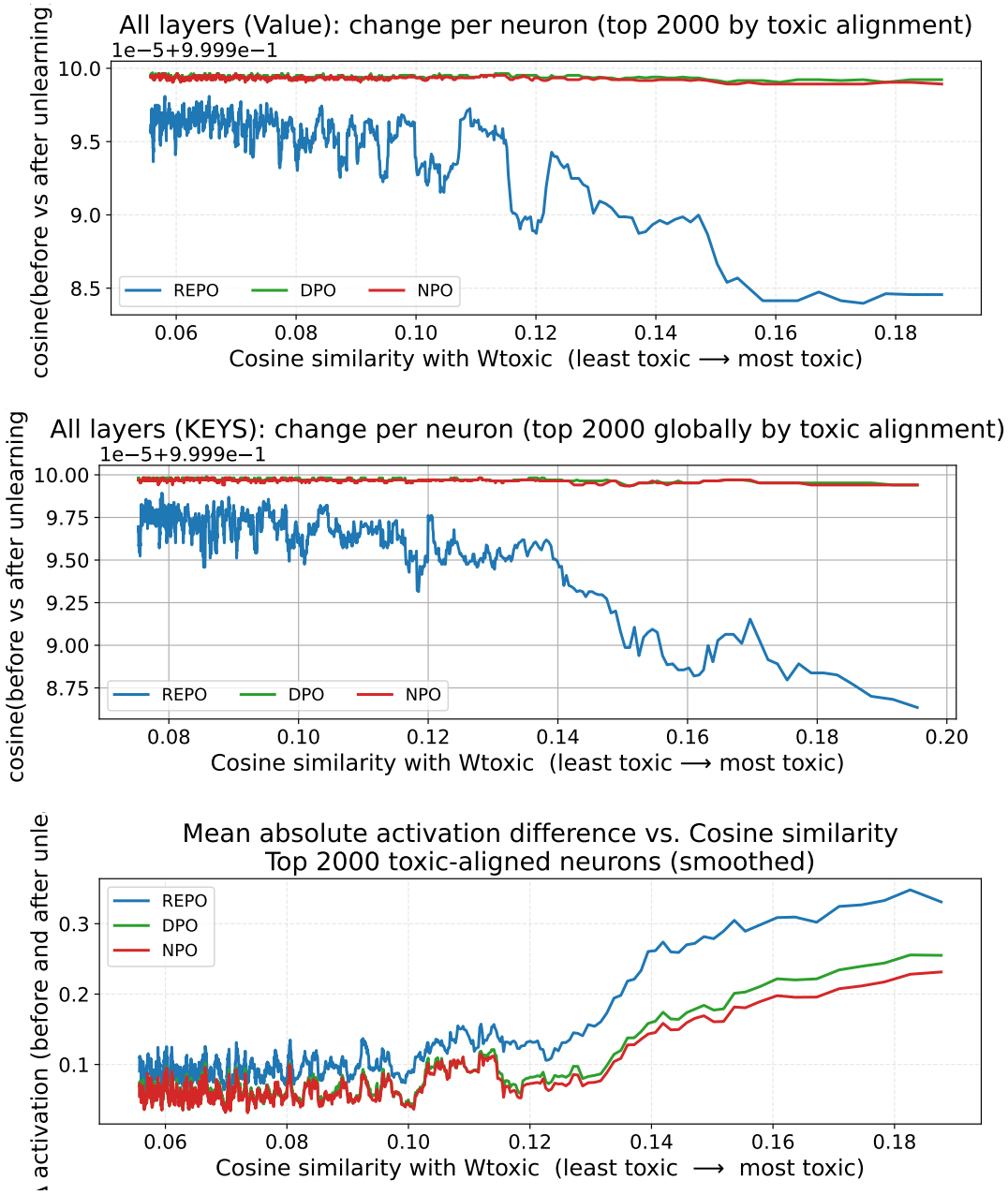

Figure 6. Comparison of how unlearning methods affect model internals. Top: cosine similarity between pre- and post-unlearning value vectors for the top 2000 toxic-aligned neurons. Middle: cosine similarity for key vectors of the top 2000 globally toxic-aligned neurons. Bottom: mean absolute activation difference vs. cosine similarity for the same neurons. Each curve shows REPO, DPO, and NPO behaviour as a function of cosine similarity with W_{toxic} (left = least toxic, right = most toxic).

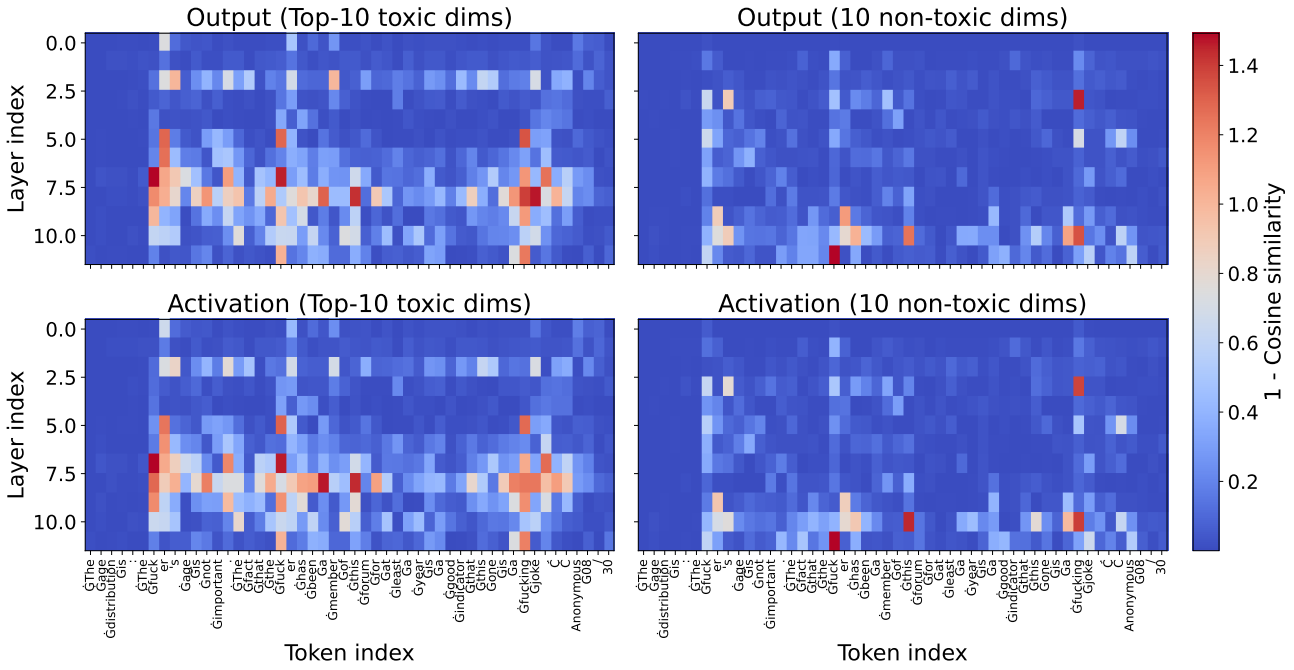

Figure 7. Layer–token residual-stream drift ($1-$cosine similarity) between the reference and REPO models for the same negative prompt. **Top:** Differences in residual contributions (post-activation keys multiplied by value vectors). **Bottom:** Differences in key activations. Within each row, **Left** shows the top-10 toxic dimensions (most aligned with W_{toxic}) and **Right** shows 10 non-toxic dimensions. Rows correspond to GPT-2 Small layers and columns to prompt tokens; darker colors indicate greater similarity and yellow larger drift.

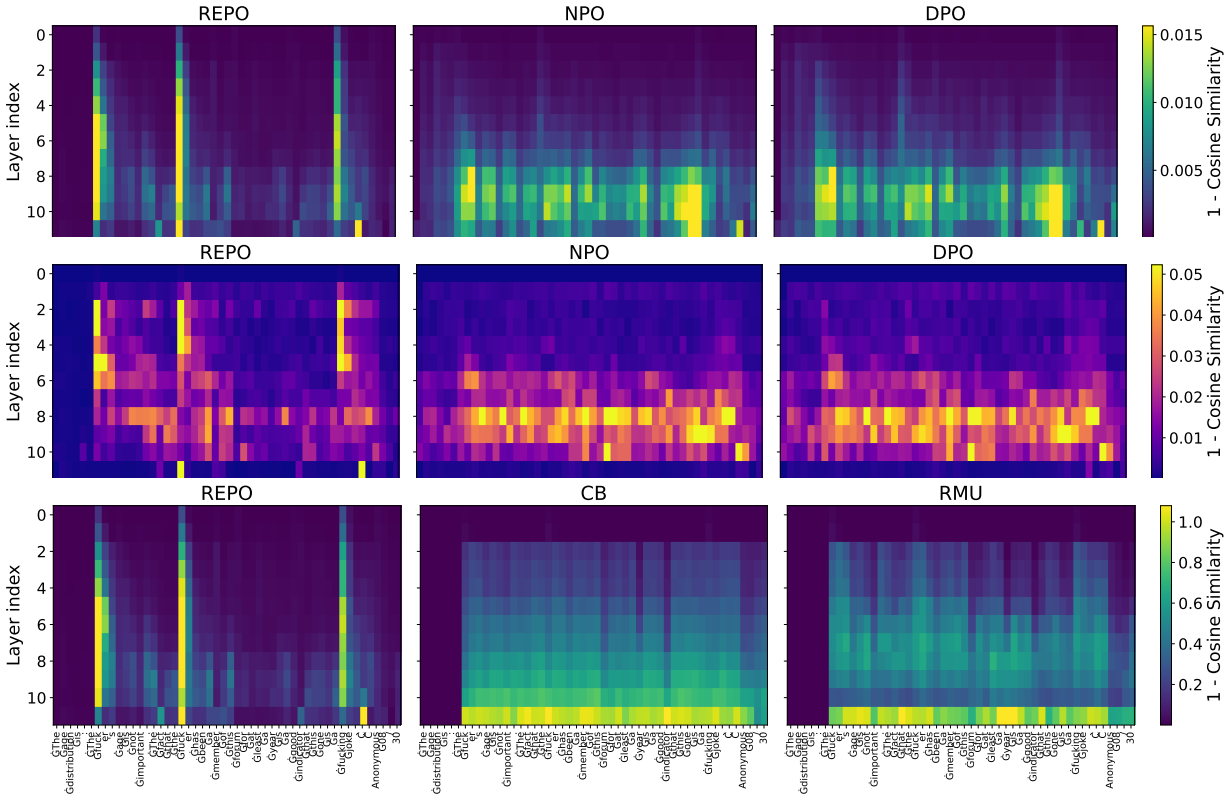

Figure 8. Layer–token distance heatmaps for different methods on a sample prompt. Columns show (left to right) REPO, NPO, and DPO (top two rows), and REPO, CB, and RMU (bottom row). **Top:** $1 - \cos$ similarity between unlearned and reference hidden states across GPT-2 small layers (y-axis) and tokens (x-axis); darker indicates higher similarity. **Middle:** $1 - \cos$ similarity between attention submodule outputs (before residual addition) of the unlearned and reference models. **Bottom:** Same as the top row, but for representation-based methods.

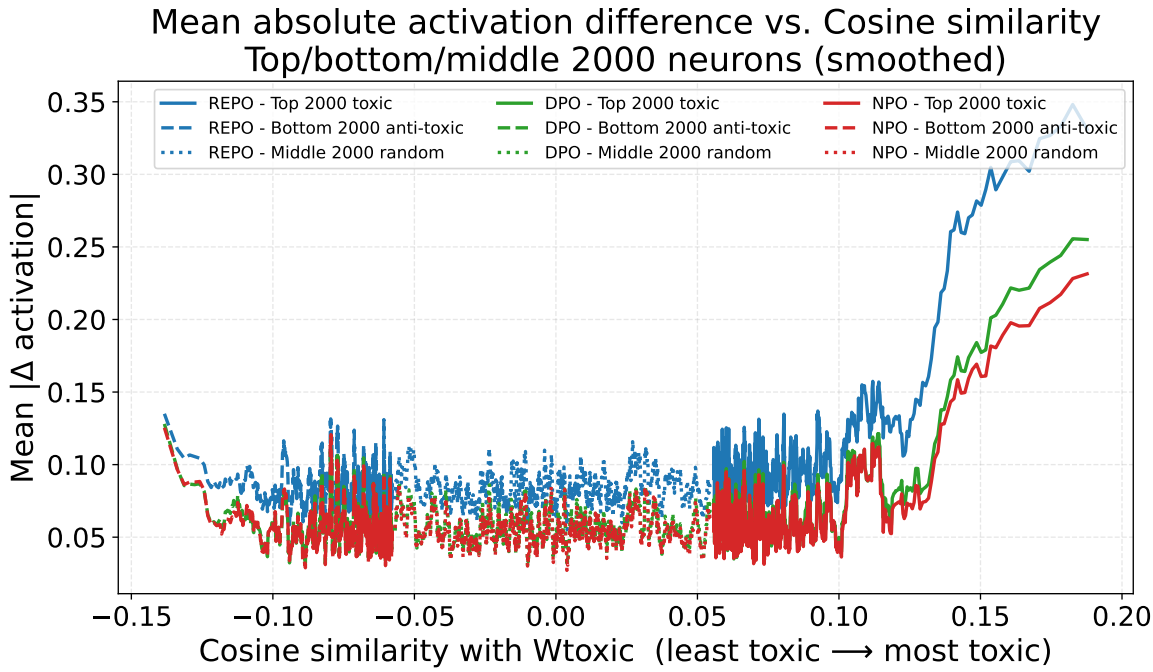

Figure 9. Mean absolute change in neuron activation vs. toxicity alignment. Each curve shows the average absolute change between the unlearned and reference models, plotted against the neuron's cosine similarity to the learned toxicity direction W_{toxic} (x-axis). Solid lines: top 2,000 aligned neurons; dashed: bottom 2,000 (anti-aligned); dotted: 2,000 random remaining neurons. Colours indicate unlearning methods (REPO, DPO, NPO). Higher y-values indicate larger deviations from the reference model; curves are smoothed with a moving window of 20.

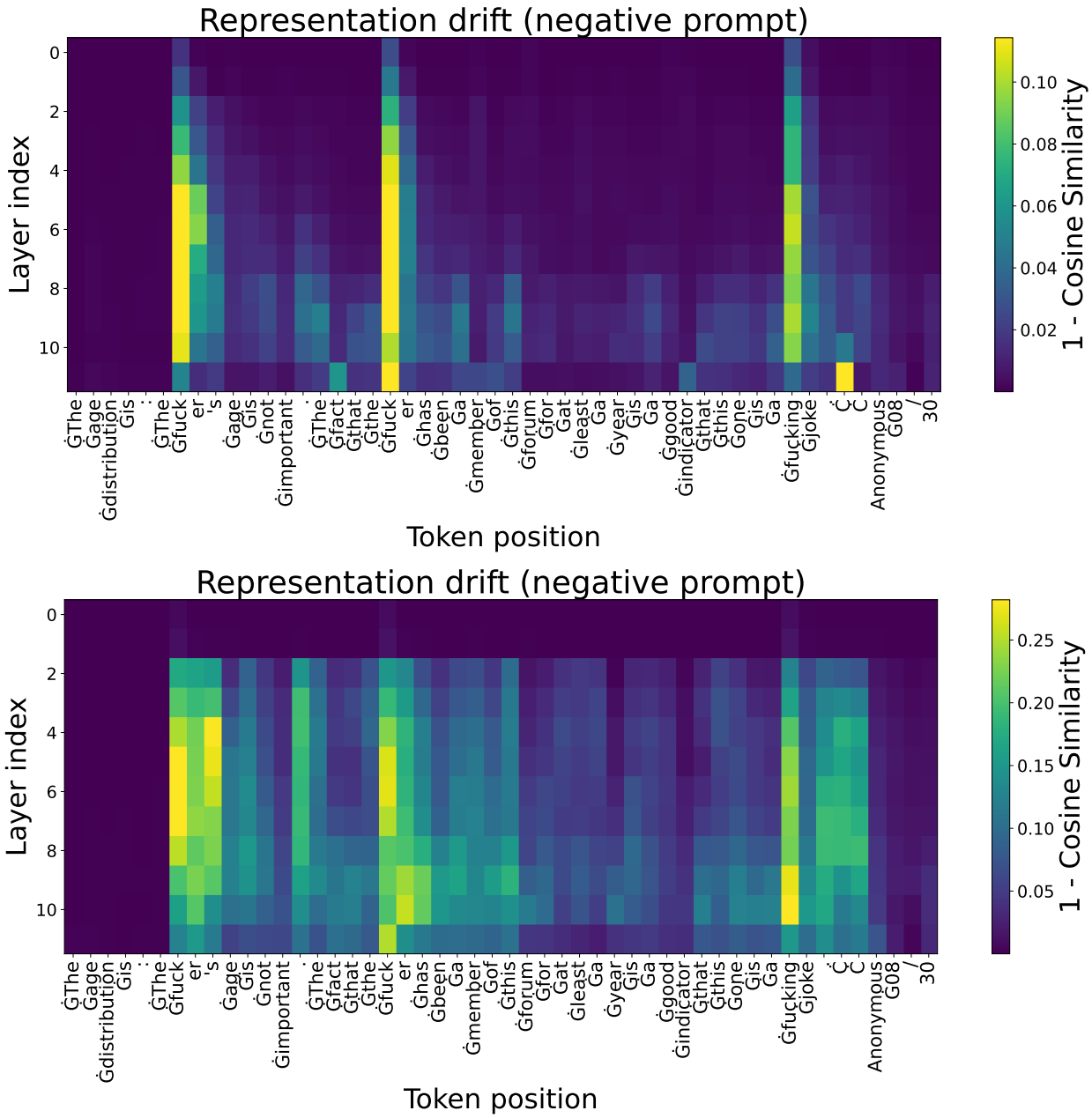

Figure 10. Layer–token representation drift (1−cosine similarity) for the same negative prompt under two discriminator input strategies in REPO: **Top** — individual tokens; **Bottom** — non-overlapping averaged segments. Darker colours indicate greater similarity, yellow larger drift.