Distributional Surgery for Language Model Activations

Bao Nguyen
CUHK
nbnguyen@se.cuhk.edu.hk

Binh Nguyen
VinUniversity
binh.nt2@vinuni.edu.vn

Duy Nguyen
UNC-Chapel Hill
duykng@cs.unc.edu

Viet Anh Nguyen
CUHK
nguyen@se.cuhk.edu.hk

Abstract

Language models, while capable of generating remarkably coherent and seemingly accurate text, can occasionally produce undesirable content including harmful or toxic outputs. In this paper, we present a new two-stage approach to detect and mitigate undesirable content generations by rectifying activations. First, we train an ensemble of layerwise classifiers to detect undesirable content using activations by minimizing a smooth surrogate of the risk-aware score. Then, for detected undesirable contents, we propose layerwise distributional steering policies that transform the attention heads. These policies are computed through principled semidefinite programming aims to minimally perturb the attention distribution while probabilistically guaranteeing the effectiveness of the editions. Empirical evaluations across multiple language models and datasets show that our method outperforms baselines in reducing the generation of undesirable output.

1 Introduction

Language models (LMs) have demonstrated a remarkable ability to understand and generate human-like documents (Radford et al., 2019; Brown et al., 2020; Touvron et al., 2023a,b; Jiang et al., 2023; Dubey et al., 2024). However, inspection of their output often reveals undesirable generation, including inaccurate or toxic texts (Ji et al., 2023; Rawte et al., 2023; Xu et al., 2024). Meanwhile, developing effective strategies to control the generation process of LMs remains a significant challenge (Tonmoy et al., 2024).

Researchers have proposed numerous methods for controllable text generation in language models (Zhang et al., 2023; Li et al., 2024a). These approaches primarily include model editing and supervised fine-tuning. Both methods, however, require altering the model weights using a subset of text samples, which can result in unstable representations for other text instances (Hase et al., 2024). Additionally, these approaches typically require substantial computational resources.

To address these limitations, *activation intervention* has emerged as a promising alternative for controllable text generation (Subramani et al., 2022; Hernandez et al., 2023; Li et al., 2024b). This approach involves altering the model activations responsible for undesirable output during inference. Previous research has identified interpretable directions within the activation space of language models that play a causal role during inference. Studies by Burns et al. (2022) and Moschella et al. (2023) suggest that these directions can be manipulated to adjust model behavior in a controlled manner. This body of work indicates that the internal representations of language models are structured in ways that enable fine-grained control over generated text.

Drawing inspiration from these findings, activation intervention frameworks operate on the premise that the information needed to guide the model toward generating a target sentence is *already encoded within the model*. The hidden information is extracted as latent vectors and then used to steer the generation toward producing desirable outputs. The preliminary success of these activation intervention methods motivates our approach to improving the quality and controllability of language model generation.

Problem Statement. We consider a language model consisting of L layers, each layer has H heads, each head has dimension d. For example, for Llama-2, we have $L = 32$, $H = 32$, and $d = 128$. The training dataset is denoted by $\mathcal{D} = (x_i, y_i^*)_{i=1,\ldots,N}$, the i-th text is denoted by x_i, and its ground truth label is $y_i^* \in \{0, 1\}$, where the label 1 (positive) represents the *un*desirable text, and the label 0 (negative) represents the desirable text. Our goal is twofold: (i) detect an undesirable text, and (ii) modify an undesirable text into a desirable text.

The activations for a text x_i at layer $\ell \in$

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 8192–8212
November 4-9, 2025

$\{1, \ldots, L\}$ is denoted by $a_{\ell,i}$. The activation at layer $\ell + 1$ is the output of the operation:

$$a_{\ell+1,i} = a_{\ell,i}^{\text{mid}} + \text{FFN}(a_{\ell,i}^{\text{mid}}),$$
$$a_{\ell,i}^{\text{mid}} = a_{\ell,i} + \sum_{i=1}^{H} Q_{\ell h} \text{Att}(P_{\ell h} a_{\ell,i}). \quad (1)$$

Here, $P_{\ell h} \in \mathbb{R}^{d \times dH}$ is the projection matrix that maps the output of each layer to the d dimensional head space, Att is the attention operator (Vaswani et al., 2017), $Q_{\ell h} \in \mathbb{R}^{dH \times d}$ is the pull-back matrix and FFN is the feed-forward layer. Each $a_{\ell,i}$ is a concatenation of headwise activations $a_{\ell h,i}$ for $h = 1, \ldots, H$. Inspired by Li et al. (2024b), we aim to perform intervention at *some selected* $a_{\ell h,i}$, *the activations for head* h *of layer* ℓ, if we detect that the activation is from undesirable content.

Contributions. We contribute an activation intervention method to detect and rectify the undesirable generation of LM. We call our method RADIANT (**R**isk-**A**ware **D**istributional **I**ntervention Policies for Language Models' **A**ctivatio**n**s). RADIANT comprises two components:

1. A layerwise probe: at each layer, we train a classifier to detect undesirable content from the layer's activations. We train a risk-aware logistic classifier for each head that balances the false positive and false negative rates. Then, we aggregate these headwise classifiers' predictions using a voting mechanism to form a layerwise classifier. We then identify one layer where the probe delivers the most reasonable predictive performance. This optimal classifier serves as the detector of undesirable content.

2. A collection of headwise interventions: given the optimal layer for the layerwise probe found previously, we find for each head in that layer an optimal headwise intervention policy. We choose a simple linear map for this intervention policy that minimizes the magnitude of editing while delivering sufficient distributional guarantees that the undesirable-predicted activations will be edited into desirable-predicted activations. We show that this linear map can be computed efficiently using semidefinite programming.

1.1 Related Works

Controllable generation. Controllable text generation methods aim to alter the outputs of large language models in a desired way. One possible approach is model editing (Wang et al., 2023a; Zhang et al., 2024), which involves modifying the parameters of a model to steer its outputs. For example, Meng et al. (2022) involves identifying specific middle-layer feedforward modules that correspond to factual knowledge and then altering these weights to correct or update the information encoded by the model. Other notable methods include fine-tuning techniques such as Supervised Fine-Tuning (SFT, Peng et al. 2023; Gunel et al. 2020) and Reinforcement Learning from Human Feedback (RLHF, Ouyang et al. 2022a; Griffith et al. 2013).

Probing. Probing is a well-established framework to assess the interpretability of neural networks (Alain and Bengio, 2016; Belinkov, 2022). Probing techniques have been applied to understand the internal representations of transformer architectures in language models such as BERT and GPT. For example, Burns et al. (2022) proposed an unsupervised probing method that optimizes consistency between positive and negative samples. Marks and Tegmark (2023) computes the mean difference between true and false statements and skews the decision boundary by the inverse of the covariance matrix of the activations.

Activation interventions. Activation intervention at inference time is an emerging technique for controllable generation (Turner et al., 2023; Li et al., 2024b; Singh et al., 2024; Yin et al., 2024). Unlike model editing or fine-tuning techniques, inference time intervention does not require altering the model parameters. Li et al. (2024b) proposed a headwise intervention method for eliciting truthful generated answers of a language model. They first train linear probes on each head of the language model, then shift the activations with the probe weight direction or the mean difference direction.

There is a clear distinction between our method and ITI when choosing the location of the classifiers and, hence, the location of the interventions. The ITI method builds different headwise classifiers scattered at *different* layers, and it may suffer from distribution shifts: if an activation is intervened, this leads to shifts in the activation values at all subsequent layers in the network. Thus, classifiers trained at subsequent layers can degrade performance, and interventions at subsequent layers can also degrade. On the contrary, we build a layerwise classifier focusing on all heads in the *same* layer and does not suffer from the distributional

shifts of the activations.

The recent paper by Singh et al. (2024) is closely related to our work. The authors propose a heuristic intervention rule; then, using empirical estimations of the means and covariances of activations data's distributions of desirable and undesirable text, they calculate a closed-form optimal transport plan between these two empirical distributions, assuming they are standard normal. However, this framework does not take into account the semantics of sentences. Another recent method, called LoFit (Localized Fine-Tuning on LLM Representations Yin et al. 2024), also identifies a specific subset of attention heads that are crucial for learning a particular task, but then performs fine-tuning on the intervention vectors at those chosen heads to enhance the model's hidden representations. This results in additional training overhead.

2 Layerwise Risk-aware Probes

In the first step, we aim to find a classifier $\mathcal{C}_{\ell h} : \mathbb{R}^d \to \{0, 1\}$ for each head $h = 1, \ldots, H$ at each layer $\ell = 1, \ldots, L$ to classify the activation value $a_{\ell h}$ of desirable and undesirable texts. We propose using a linear logistic classifier, parameterized by a slope parameter $\theta_{\ell h} \in \mathbb{R}^d$ and a bias parameter $\vartheta_{\ell h} \in \mathbb{R}$. The headwise classification rule is

$$\mathcal{C}_{\ell h}(a_{\ell h}) = \begin{cases} 1 & \text{if } \mathrm{sigmoid}(\vartheta_{\ell h} + \theta_{\ell h}^\top a_{\ell h}) \geq 0.5, \\ 0 & \text{otherwise,} \end{cases}$$
$$= \begin{cases} 1 & \text{if } \vartheta_{\ell h} + \theta_{\ell h}^\top a_{\ell h} \geq 0, \\ 0 & \text{if } \vartheta_{\ell h} + \theta_{\ell h}^\top a_{\ell h} < 0. \end{cases}$$

The training process of $\mathcal{C}_{\ell h}$ must take into account two types of risk: (i) false-negative risk when an undesirable text is not detected, (ii) false-positive risk when a desirable text is classified as undesirable, and is subsequently edited and loses its original semantics. Therefore, a natural candidate for the loss function is a combination of the False Positive Rate (FPR) and the False Negative Rate (FNR). However, neither FPR nor FNR have smooth functions in optimizing variables. We, hence, resort to smooth surrogates of these two metrics that use the predicted probability of the classifier, similarly to Bénédict et al. (2022). In detail, we use

$$\mathrm{FPR}(\theta_{\ell h}, \vartheta_{\ell h}) = \frac{1}{N_0} \sum_{i=1}^{N} \mathrm{sigmoid}(\vartheta_{\ell h} + \theta_{\ell h}^\top a_{\ell h,i}) \times (1 - y_i^*),$$
$$\mathrm{FNR}(\theta_{\ell h}, \vartheta_{\ell h}) = \frac{1}{N_1} \sum_{i=1}^{N} \Big(1 - \mathrm{sigmoid}(\vartheta_{\ell h} + \theta_{\ell h}^\top a_{\ell h,i})\Big) \times y_i^*.$$

The linear probe training loss is thus

$$\min_{\theta_{\ell h} \in \mathbb{R}^d,\ \vartheta_{\ell h} \in \mathbb{R}} \mathrm{FPR}(\theta_{\ell h}, \vartheta_{\ell h}) + \alpha \mathrm{FNR}(\theta_{\ell h}, \vartheta_{\ell h}), \tag{2}$$

for some positive weight parameters α. A higher value of α will emphasize achieving a lower false negative rate, which is critical for detecting undesirable inputs. Problem (2) has a smoothed surrogate loss that is differentiable and can be solved using a gradient descent algorithm. Finally, we aggregate $\{\mathcal{C}_{\ell h}\}_{h=1,\ldots,H}$ into a single classifier $\mathcal{C}_\ell$ for layer ℓ by a simple voting rule

$$\mathcal{C}_\ell(a_\ell) = \begin{cases} 1 & \text{if } \sum_{h=1}^{H} \mathcal{C}_{\ell h}(a_{\ell h}) \geq \tau, \\ 0 & \text{otherwise,} \end{cases}$$

where $\tau \in [0, H]$ is a tunable threshold. When $\tau = \lfloor H/2 \rfloor$, then $\mathcal{C}_\ell$ becomes the majority voting results of the individual (weak) classifiers $\mathcal{C}_{\ell h}$. We optimize the hyperparameter τ to reduce the False Negative Rate (FNR), with a secondary focus on the False Positive Rate (FPR) in cases of equal FNR rates. The rationality for this choice is that we believe undesirable content being labeled as desirable is more problematic than other instances.

To conclude this step, we compute the classifier $\mathcal{C}_\ell$ for layer $\ell = 1, \ldots, L$ by tuning the parameters α. The layer whose classifier $\mathcal{C}_\ell$ delivers the highest quality (accuracy or any risk-aware metric) will be the optimal layer to construct the probe. This optimal layer, along with the collection of headwise classifiers, is the final output of this step.

3 Headwise Interventions with Probabilistic Guarantees

We propose a distributional intervention to the activations of the samples predicted as undesirable by the layerwise classifier. In this section, we will focus on constructing a single headwise intervention, and in the next section, we will combine multiple headwise interventions into a layerwise

intervention. A headwise intervention is a map $\Delta_{\ell h} : a_{\ell h} \mapsto \hat{a}_{\ell h}$ that needs to balance multiple criteria: (i) it should be easy to compute and deploy, (ii) it should be effective in converting the undesirable activations to the desirable regions, (iii) it should minimize the magnitude of the intervention to sustain the context of the input. Intuitively, we propose solving an optimization problem with the loss and constraints that fit all the criteria listed. The details are as follows.

To promote (i), we employ a simple linear map $\Delta_{\ell h}(a_{\ell h}) = G_{\ell h} a_{\ell h} + g_{\ell h}$ parametrized by a matrix $G_{\ell h} \in \mathbb{R}^{d \times d}$ and a vector $g_{\ell h} \in \mathbb{R}^d$. This linear map can also be regarded as a pushforward map that transforms the *un*desirable-predicted activations to become desirable-predicted activations. Let us now represent the *un*desirable-predicted activations as a d-dimensional random vector $\tilde{a}_{\ell h}$. Its distribution can be estimated using the training data after identifying the subset $\hat{\mathcal{D}}^+_{\ell h}$ of training samples that are *predicted undesirable* by $\mathcal{C}_{\ell h}$, that is, $\hat{\mathcal{D}}^+_{\ell h} \triangleq \{i : \mathcal{C}_{\ell h}(a_{\ell h,i}) = 1\}$. The activations of samples in $\hat{\mathcal{D}}^+_{\ell h}$ lead to an empirical distribution $\widehat{\mathbb{P}}_{\ell h}$. The linear map $\Delta_{\ell h}$ will pushforward the distribution $\widehat{\mathbb{P}}_{\ell h}$ to the new distribution $\mathbb{Q}_{\ell h} = \Delta_{\ell h} \# \widehat{\mathbb{P}}$.

Using the pushforward distribution $\mathbb{Q}_{\ell h}$, we can impose criteria (ii) and (iii) above in an intuitive method. To promote (ii), we require that the activations distributed under $\mathbb{Q}_{\ell h}$ should be classified as desirable by $\mathcal{C}_{\ell h}$ with high probability. Finally, to promote (iii), we require that the distributions $\mathbb{Q}_{\ell h}$ and $\widehat{\mathbb{P}}_{\ell h}$ be not too far from each other. Let $\gamma \in (0, 0.5)$ be a small tolerance parameter, and let φ be a measure of dissimilarity between probability distributions, we propose to find $\Delta_{\ell h}$ by solving the following stochastic program

$$\begin{array}{ll} \min & \varphi(\widehat{\mathbb{P}}_{\ell h}, \mathbb{Q}_{\ell h}) \\ \text{s.t.} & \mathbb{Q}_{\ell h}(\tilde{a} \text{ classified by } \mathcal{C}_{\ell h} \text{ as } 0) \geq 1 - \gamma, \\ & \mathbb{Q}_{\ell h} = \Delta_{\ell h} \# \widehat{\mathbb{P}}_{\ell h}. \end{array} \tag{3}$$

Problem (3) is easier to solve in specific circumstances. For example, when we impose that both $\widehat{\mathbb{P}}_{\ell h}$ and $\mathbb{Q}_{\ell h}$ are Gaussian and when we choose φ as a moment-based divergence, then $\Delta_{\ell h}$ can be obtained by solving a convex optimization problem. In the next result, we use $\|\cdot\|_F$ as the Frobenius norm of a matrix, and Φ as the cumulative distribution function of a standard Gaussian distribution.

Theorem 1 (Optimal headwise intervention)**.** *Suppose that* $\widehat{\mathbb{P}}_{\ell h} \sim \mathcal{N}(\widehat{\mu}, \widehat{\Sigma})$ *and* $\mathbb{Q}_{\ell h} \sim \mathcal{N}(\mu, \Sigma)$ *and*

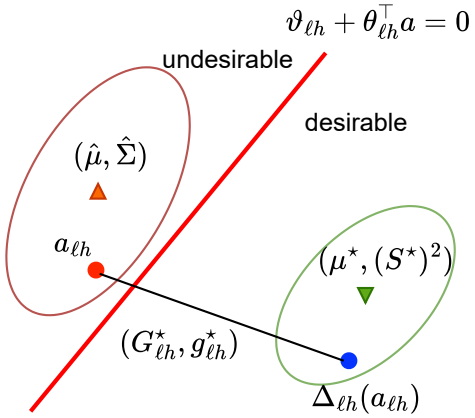


Figure 1: At head h of layer ℓ, we learn a headwise intervention, linear mapping $\Delta_{\ell h}$ to transform the *un*desirable-predicted activations to desirable-predicted activations.

φ admits the form

$$\varphi(\widehat{\mathbb{P}}_{\ell h}, \mathbb{Q}_{\ell h}) = \|\mu - \widehat{\mu}\|_2^2 + \|\Sigma^{\frac{1}{2}} - \widehat{\Sigma}^{\frac{1}{2}}\|_F^2.$$

Let $(\mu^\star, S^\star, t^\star)$ be the solution of the following semidefinite program

$$\begin{array}{ll} \min & \|\mu - \widehat{\mu}\|_2^2 + \|S - \widehat{\Sigma}^{\frac{1}{2}}\|_F^2 \\ \text{s.t.} & \vartheta_{\ell h} + \theta_{\ell h}^\top \mu + \Phi^{-1}(1-\gamma) t \leq 0 \\ & \|S \theta_{\ell h}\|_2 \leq t \\ & \mu \in \mathbb{R}^d,\ S \in \mathbb{S}_+^d,\ t \in \mathbb{R}_+. \end{array} \tag{4}$$

Then, by defining $G^\star_{\ell h} = \widehat{\Sigma}^{-\frac{1}{2}} \big(\widehat{\Sigma}^{\frac{1}{2}} (S^\star)^2 \widehat{\Sigma}^{\frac{1}{2}}\big)^{\frac{1}{2}} \widehat{\Sigma}^{-\frac{1}{2}}$ and $g^\star_{\ell h} = \mu^\star - G^\star_{\ell h} \widehat{\mu}$, a linear map $\Delta_{\ell h}$ that solves (3) is

$$\Delta_{\ell h}(a_{\ell h}) = G^\star_{\ell h} a_{\ell h} + g^\star_{\ell h}.$$

The proof of Theorem 1 can be found in Appendix D. Figure 1 illustrates the effect of the headwise intervention $\Delta_{\ell h}$: The headwise classifier $\mathcal{C}_{\ell h}$ is represented by the red linear hyperplane $\vartheta_{\ell h} + \theta_{\ell h}^\top a = 0$ on the activation space; the undesirable-predicted (label 1) region is towards the top left corner, while the desirable-predicted (label 0) region is towards the bottom right corner. The activations of the undesirable-predicted samples are represented as a Gaussian distribution with mean $(\widehat{\mu}, \widehat{\Sigma})$, drawn as the red ellipsoid. The edit map $\Delta_{\ell h}$ pushes this distribution to another Gaussian distribution $\mathbb{Q}_{\ell h}$ drawn as the green ellipsoid. The distribution $\mathbb{Q}_{\ell h}$ has a coverage guarantee on the desirable-predicted region with probability at least $1 - \gamma$. One can also verify that $\mathbb{Q}_{\ell h}$ has mean $\mu^\star$ and covariance matrix $(S^\star)^2$. Problem (4) can be solved using semidefinite programming solvers such as COPT or Mosek.

The moment information $\widehat{\mu}$ and $\widehat{\Sigma}$ can be estimated from the subset $\hat{\mathcal{D}}^{+}_{\ell h}$. One can intuitively expect a trade-off between the tolerance level γ and the magnitude of the headwise mapping. If γ is lower, the activations will be edited at a higher magnitude so that the edited activations will likely end up in the desirable-predicted region of the classifier $\mathcal{C}_{\ell h}$. In contrast, if γ is higher, the activations will be edited with a smaller magnitude due to the less stringent constraint to swap the predicted label.

One can view the distribution $\mathbb{Q}_{\ell h} \sim (\mu^\star, (S^\star)^2)$ as the counterfactual distribution of the undesirable-predicted activations with *minimal* perturbation. This distribution $\mathbb{Q}_{\ell h}$ is found by optimization, which is in stark contrast with the design of the counterfactual distribution in MiMic (Singh et al., 2024), in which the intervention is computed based on the activations of the desirable-predicted activations. As a comparison to ITI (Li et al., 2024b), we note that the headwise intervention of ITI does *not* depend on the value of the activations: ITI shifts the activations along the truthful directions for a stepsize multiplied by the standard deviation of activations along the intervention (truthful) direction. In contrast, our headwise intervention depends on the value $a_{\ell h}$, and one can verify that the magnitude of the proposed shift amounts to $\|(G^\star_{\ell h} - I)a_{\ell h} + g^\star_{\ell h}\|_2$. Moreover, ITI does not provide any (probabilistic) guarantee for the intervention, while the probabilistic guarantee is internalized in our method through the design of the mapping in equation (3).

Remark 1. *We observe that the two following tricks increase the empirical performance of our intervention framework. First, to avoid the collapse of $\mathbb{Q}_{\ell h}$ into a Dirac distribution and to ensure the similarity between the real and the constructed covariance matrix of desirable content, we can add the constraint $S \succeq \widehat{\Sigma}_0^{\frac{1}{2}}$ to the optimization problem (4), where $\widehat{\Sigma}_0$ is the empirical covariance matrix of the desirable activations $\{i : y_i^* = 0\}$. Second, to avoid taking the inverse cdf of the standard normal distribution, we use $\Gamma \leftarrow \Phi^{-1}(1-\gamma)$ and finetune Γ instead of γ.*

Finally, given the input with activation a_ℓ at layer ℓ, suppose that a_ℓ is predicted undesirable by $\mathcal{C}_\ell$, we propose to edit the activations of *only* the heads that are predicted undesirable by the headwise classifier $\mathcal{C}_{\ell h}$. More specifically, we edit the headwise activations $a_{\ell h}$ to a new headwise activations $\hat{a}_{\ell h}$ through the relationship

$$\hat{a}_{\ell h} = \begin{cases} \Delta_{\ell h}(a_{\ell h}) & \text{if } \mathcal{C}_{\ell h}(a_{\ell h}) \times \mathcal{C}_\ell(a_\ell) = 1, \\ a_{\ell h} & \text{otherwise,} \end{cases}$$

where $\Delta_{\ell h}(a_{\ell h}) = G^\star_{\ell h} a_{\ell h} + g^\star_{\ell h}$ for $h = 1, \ldots, H$. In other words, each new headwise activation $\hat{a}_{\ell h}$ is computed based on three terms: the original headwise activations $a_{\ell h}$, the headwise intervention $\Delta_h(a_{\ell h})$, and the indicator value identifying if head h and layer ℓ is predicted desirable or undesirable.

4 Experiments

In this section, we present empirical evidence for the effectiveness of our method RADIANT. We evaluate RADIANT on the TruthfulQA benchmark Lin et al. (2021), consisting of two tasks: the main task is the generation, and the secondary task is multiple choice. The generation task requires the model to generate an entire answer for each question using greedy autoregressive decoding. The accuracy and helpfulness of the answer are best assessed by humans. However, in almost all recent works in the field, including Li et al. (2024b) and Yin et al. (2024), this criterion is measured by an alternative large language model finetuned on the target dataset. The multiple-choice task contains candidate answers to each question, requiring the model to give probabilities. Higher probabilities for truthful answers yield higher scores.

4.1 Experimental Settings

Datasets. We evaluate and compare our method with other baselines using the TruthfulQA benchmark Lin et al. (2021). Details about this dataset and how we preprocess the data can be found in Appendix A.1. In addition, we also show the generalization of our method by conducting a transferability experiment on two other out-of-distribution datasets, including NQOpen (Kwiatkowski et al., 2019) and TriviaQA (Joshi et al., 2017). Due to space constraints, the results for the latter two datasets are relegated to Appendix A.5.

Hyperparameters. There are two pivotal hyperparameters in the RADIANT framework, namely α in probe loss (2), and $\Gamma = \Phi^{-1}(1-\gamma)$ in the computation of the intervention map (4). The discussion about their impact on RADIANT and how to select them is in Appendix A.2.3.

Baselines. We benchmark against:

- Inference-time Intervention (ITI) (Li et al., 2024b), the state-of-the-art method for

finetuning-free intervention. The hyperparameters of the baseline follow their original paper and their GitHub repository.[1]

- Few-shot prompting (FSP) introduced in Bai et al. (2022) showcases the effectiveness of 50-shot prompting in benchmark TruthfulQA.

- Probe-Free Low Rank Activation Intervention (FLORAIN) (Jiang et al., 2025) which trains a low rank component for intervening into one specified hidden representation.

- Instruction Fine-Tuning (IFT, Wang et al. 2022; Chung et al. 2024) is a popular fine-tuning approach to boost the truthfulness of language models. Two notable pretrained models in this direction, namely Alpaca-7B (Taori et al., 2023) and Vicuna-7B (Chiang et al., 2023), are adopted for comparison.

- Representation Intervention Fine-tuning (RIFT) methods aim to adjust language model activations for improved truthfulness. However, they add extra parameters and require extensive computational resources for fine-tuning. We consider LOFiT (Yin et al., 2024) for comparison.

- Non-Linear Inference Time Intervention (NL-ITI) (Hoscilowicz et al., 2024) extends ITI by introducing a non-linear multi-token probing and multi-token intervention method.

- Learnable Intervention for Truthfulness Optimization (LITO) (Bayat et al., 2024) explores a sequence of model generations based on increasing levels of intervention magnitude and then selects the most accurate response.

Metrics. Following the standard benchmark in TruthfulQA (Lin et al., 2021; Li et al., 2024b), we use the below metrics:

- For the multiple choice task, we use MC1 and MC2 metrics as defined in Lin et al. (2021). MC1 measures the model's accuracy in selecting the correct answer from the given choices, where selection is based on the highest log-probability score assigned to each completion. MC2 is the normalized total probability assigned to the set of true answers.

- For the generation task, we use two fine-tuned `GPT-3.5-instruct` models to classify whether an answer is true or false and informative or not. We report two metrics from Li et al. (2024b): truthful score True (%) and True*Info (%), a product of scalar truthful and informative score. We note that there are discrepancies between the results of ITI reproduced in our work and the original results reported in Li et al. (2024b), as the original paper used `GPT-3` based models to score these two metrics; however, at the time this paper is written, `GPT-3` is no longer available on the OpenAI platform.

- We assess how our method and baselines alter the original generation distribution using two extra metrics: the Kullback-Leibler (KL) divergence and Cross-Entropy (CE) loss of the model's next-token prediction distribution before and after intervention. Due to limited space in the main paper, these metrics for our method and baselines are detailed in Appendix A.3.

Computing resources. We run all experiments on 4 NVIDIA RTX A5000 GPUs, an i9 14900K CPU, and 128GB RAM. The semidefinite programs (4) are solved using Mosek 10.1; the average solving time for each instance is around 50 seconds.
Our repository: https://github.com/nguyenngocbaocmt02/OT-Intervention.

4.2 Numerical Results

4.2.1 Comparison between Finetuning-free Techniques

We benchmark two fine-tuning-free baselines (ITI and FSP) along with our framework RADIANT on Llama-7B, Llama3-8B, and Llama2-chat-13B with the TruthfulQA dataset. The results are presented in the **first three big rows** of Table 1. Across the three models, the combined method of FSP + RADIANT consistently achieved the highest scores in metrics such as True * Info and True, with 49% for Llama-7B, 44% for Llama3-8B, and 65% for Llama2-chat-13B. When running alone, our method, RADIANT, also demonstrated significant improvements, particularly in Llama2-chat-13B, where it achieved a True * Info score of 64% and a Truthful score of 74%. This suggests the efficiency of our framework compared with other baselines, including the current state-of-the-art ITI.

[1] https://github.com/likenneth/honest_llama/tree/master

Model	Methods	True * Info (%) ↑	True (%) ↑	MC1 ↑	MC2 ↑
Llama-7B	Unintervened	21.15	22.16	25.58	40.54
	ITI	26.52	28.03	27.78	43.59
	FSP	36.13	39.78	**34.03**	**50.34**
	NL-ITI	29.06	38.04	32.97	45.69
	LITO	39.08	41.22	29.22	47.64
	FLORAIN	31.46	34.72	31.76	47.43
	RADIANT (ours)	**40.36**	**44.48**	30.91	46.13
	FSP + ITI	40.63	45.16	35.50	52.48
	FSP + NL-ITI	45.97	47.31	**38.37**	53.61
	FSP + LITO	49.05	55.68	36.23	54.92
	FSP + FLORAIN	45.31	49.23	36.45	54.27
	FSP + RADIANT (ours)	**49.31**	**57.43**	37.97	**55.31**
Llama3-8B	Unintervened	32.88	44.18	30.36	48.98
	ITI	35.92	46.88	32.07	49.84
	FSP	36.32	39.78	**35.74**	52.93
	NL-ITI	35.98	45.72	33.02	51.37
	LITO	37.53	48.20	34.96	52.54
	FLORAIN	36.78	48.67	34.56	**53.68**
	RADIANT (ours)	**37.78**	**50.82**	33.82	52.98
	FSP + ITI	40.63	45.16	35.50	52.98
	FSP + NL-ITI	40.70	46.03	34.15	53.35
	FSP + LITO	43.95	49.82	**38.41**	55.31
	FSP + FLORAIN	42.15	47.32	36.98	**55.83**
	FSP + RADIANT (ours)	**44.09**	**52.02**	37.98	54.61
Llama2-chat-13B	Unintervened	51.87	59.86	35.38	53.32
	ITI	57.02	63.04	37.46	55.59
	FSP	55.97	58.63	**40.76**	57.84
	NL-ITI	57.13	60.82	39.01	57.24
	LITO	58.12	61.36	38.25	57.21
	FLORAIN	60.68	67.70	39.65	**59.57**
	RADIANT (ours)	**63.68**	**74.20**	39.95	58.18
	FSP + ITI	56.78	59.24	41.50	59.01
	FSP + NL-ITI	59.62	61.77	42.15	57.87
	FSP + LITO	60.74	63.21	41.28	58.46
	FSP + FLORAIN	61.14	62.45	**44.52**	**61.48**
	FSP + RADIANT (ours)	**64.68**	**67.75**	42.52	59.99
Alpaca	Base	30.39	30.85	26.56	41.63
	+ ITI	37.67	38.19	28.89	45.19
	+ RADIANT (ours)	**44.51**	**45.94**	**30.79**	**47.83**
Vicuna	Base	38.24	42.10	31.83	48.48
	+ ITI	49.27	53.25	33.42	51.80
	+ RADIANT (ours)	**54.87**	**62.81**	**35.76**	**55.14**
Llama variants + LOFiT	LOFiT (7B)	59.48	69.03	51.04	70.78
	+ ITI	60.84	**72.29**	51.41	70.84
	+ RADIANT (ours)	**61.50**	72.08	**51.80**	**71.29**
	LOFiT (8B)	68.80	90.08	59.00	**77.93**
	+ ITI	67.57	79.31	55.33	75.85
	+ RADIANT (ours)	**71.47**	**90.19**	**59.30**	76.56
	LOFiT (Chat-13B)	66.35	81.89	57.04	**76.17**
	+ ITI	66.00	78.09	55.08	75.25
	+ RADIANT (ours)	**69.63**	**83.86**	**57.45**	75.47

Table 1: Quantitative results of different intervention methods on TruthfulQA dataset, across different Language Models and fine-tuning approaches. Parameters of RADIANT: $\alpha = 2.5, \Gamma = 15$.

4.2.2 Comparison against ITI with Instruction Finetuning Methods.

We investigate whether implementing RADIANT on Alpaca and Vicuna, two instruction fine-tuning models from Llama-7B, can further enhance their performances. Results in Table 1 (**fourth and fifth big rows**) indicate that applying RADIANT significantly enhances both the baseline models, with Alpaca + RADIANT improved to 44.5% in True*Info score and 46% in Truthful score. Similarly, Vicuna + RADIANT achieved the highest scores of 55% in True*Info score and 63% in Truthful score, showcasing a marked increase compared to its baseline performance of 38% and 42.1%, respectively. In both cases, RADIANT outperformed ITI, demonstrating its effectiveness in enhancing the models' accuracy and truthfulness.

4.2.3 Comparison against ITI with Representation Intervention Finetuning Methods.

We apply RADIANT and ITI on Llama-7B, Llama3-8B, and Llama2-chat-13B models, which were previously fine-tuned by LOFiT, a representation intervention finetuning method. The experimental results in the **last big row** of Table 1 show that RADIANT is better than ITI in improving correctness and informativeness across different Llama models. While ITI offers modest improvements in some instances, it generally lags behind RADIANT, especially in larger models.

4.3 Ablation Studies

We perform two ablation studies to demonstrate the effectiveness of our framework. Table 2 reports the performance of the Llama-7B + TruthfulQA dataset. In the first scenario, we select intervened heads using ITI, then compare our intervention approach versus ITI. We noticed that switching the head selection between RADIANT and ITI improved performance when the RADIANT intervention was applied, reaching 37% in the True * Info score. In the second scenario, the probing loss function is replaced by the popular binary cross-entropy loss. This scenario tests the impact of replacing the risk-aware loss function with cross-entropy loss, which resulted in moderate improvements but still fell short compared to RADIANT's risk-aware loss in Section 2 (30.36% vs 40.36% in True*Info). Overall, these findings suggest that both the choice of intervention and the loss function play crucial roles in our framework.

Methods	True * Info (%) ↑	True (%) ↑	MC1 ↑	MC2 ↑
Unintervened	21.15	22.16	25.58	40.54
ITI	26.52	28.03	27.78	43.59
1st scenario: Our linear probe + ITI intervention	26.88	28.00	29.00	44.00
1st scenario: ITI linear probe + our intervention	36.66	39.00	28.00	43.00
2nd scenario: Cross entropy loss	30.36	33.00	29.00	43.00
RADIANT	**40.36**	**44.48**	**30.91**	**46.13**

Table 2: Ablation study results: in the first scenario, we swap heads selected by RADIANT with ITI intervention, and vice versa; in the second scenario, we replace our risk-aware loss function with cross-entropy loss in training linear probe. Performed on TruthfulQA with Llama-7B.

Component	Llama-7B	Llama3-8B	Llama2-chat-13B
Train the linear probe for one layer (s)	15.64	17.32	29.42
Compute intervention for one head (s)	52.33	58.43	55.67
Avg. increase in inference time per answer (%)	3.09	3.32	4.72

Table 3: Wall-clock time breakdown by components of RADIANT for different pretrained models.

4.4 Computational Cost

Our method is computationally cheap: for each head, our linear probes require one vector-vector multiplication, and our linear interventions require only one matrix-vector multiplication. To demonstrate this, we clocked the running time to calculate the intervention vectors on an A5000 GPU for the Llama-7B and Llama3-8B models and on two A5000 GPUs for Llama2-chat-13B and show the results in Table 3. Our intervention only slightly increases the running time of the inference process. In addition to its simplicity, the preprocessing of our framework for calculating intervention vectors is much less time-consuming and resource-intensive than fine-tuning methods.

4.5 Additional Experiments

We perform multiple additional benchmarks to demonstrate the effectiveness of RADIANT across diverse settings, but due to the lack of space, we put the rest of these experiments in the Appendix. More specifically:

- We conduct a hyperparameter analysis and compare our risk-aware loss function with weighted negative log-likelihood (Section A.2). We also report the results for the Kullback-Leibler (KL) and Cross-Entropy (CE) metric for generation (Section A.3). We find consistent improvements of RADIANT compared to other competitors.
- We develop a parallel implementation (RADIANT-P) in Section A.4 that significantly reduces computational overhead while maintaining the performance of the intervention.
- We conduct a transferability analysis (Section A.5) to demonstrate that RADIANT has better generalization to the NQOpen and TriviaQA datasets than ITI.
- We test RADIANT on various model architectures, including Gemma and GPT models (Section B.1), Mistral and Qwen models (Section B.2), and sparse MoE architectures (Section B.3). We also compare RADIANT with supervised fine-tuning (Section B.4).
- We evaluate RADIANT on other NLP tasks, including toxicity mitigation (Section C.1), long fact generation (Section C.2), and creative writing (Section C.3). RADIANT shows comparable performance to computationally intensive methods while maintaining better fluency.

5 Conclusion

We introduced RADIANT, an activation intervention method consisting of two components: (i) a layerwise probe to detect undesirable content and (ii) headwise interventions to rectify the head activations upon undesirably predicted outcomes. Contrary to existing intervention methods, where the interventions can be scattered across different layers, our intervention is focused on a single layer of the network. This focus helps alleviate the distributional shifts of the activations in subsequent layers. Moreover, our headwise intervention aims to minimize the perturbations to the activations while keeping a reasonable guarantee of the effectiveness of the intervention. This is further demonstrated in empirical results, where our method outperforms the baseline intervention methods for various LMs.

Limitations and Social Impact. Our paper focuses on improving the truthfulness of LMs, and the results aim to improve trustworthy artificial intelligence. Apart from language generation, our paper can also be implemented in other domains for activation editing. However, it is important to acknowledge the potential misuse of our method. There exists a risk that adversarial actors could exploit our approach to transform truthful outputs into misleading or false information. This dual-use nature underscores the importance of ethical guidelines and safeguards in developing artificial intelligence. By promoting transparency and accountability in using our framework, we want to raise awareness of the risks while maximizing the benefits of improved truthfulness in language generation.

References

Guillaume Alain and Yoshua Bengio. 2016. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Farima Fatahi Bayat, Xin Liu, H Jagadish, and Lu Wang. 2024. Enhanced language model truthfulness with learnable intervention and uncertainty expression. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 12388–12400.

Yonatan Belinkov. 2022. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219.

Gabriel Bénédict, Hendrik Vincent Koops, Daan Odijk, and Maarten de Rijke. 2022. sigmoidF1: A smooth F1 score surrogate loss for multilabel classification. *Transactions on Machine Learning Research*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901.

Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. 2022. Discovering latent knowledge in language models without supervision. *arXiv preprint arXiv:2212.03827*.

Yi Cheng, Yilun Wu, Adrian Weller, Matt J. Kusner, and Pushmeet Kohli. 2024. Integrative decoding: Improving factuality in large language model generation via implicit self-consistency. In *International Conference on Learning Representations*.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. 2023. Vicuna: An open-source chatbot impressing GPT-4 with 90% ChatGPT quality.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2019. Plug and play language models: A simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*.

Abhimanyu Dubey et al. 2024. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A Smith. 2020. Realtoxicityprompts: Evaluating neural toxic degeneration in language models. *arXiv preprint arXiv:2009.11462*.

Shane Griffith, Kaushik Subramanian, Jonathan Scholz, Charles L Isbell, and Andrea L Thomaz. 2013. Policy shaping: Integrating human feedback with reinforcement learning. *Advances in Neural Information Processing Systems*, 26.

Beliz Gunel, Jingfei Du, Alexis Conneau, and Ves Stoyanov. 2020. Supervised contrastive learning for pre-trained language model fine-tuning. *arXiv preprint arXiv:2011.01403*.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. *arXiv preprint arXiv:2004.10964*.

Peter Hase, Mohit Bansal, Been Kim, and Asma Ghandeharioun. 2024. Does localization inform editing? surprising differences in causality-based localization vs. knowledge editing in language models. *Advances in Neural Information Processing Systems*, 36.

Evan Hernandez, Belinda Z Li, and Jacob Andreas. 2023. Measuring and manipulating knowledge representations in language models. *arXiv preprint arXiv:2304.00740*.

Jakub Hoscilowicz, Adam Wiacek, Jan Chojnacki, Adam Cieslak, Leszek Michon, and Artur Janicki. 2024. Non-linear inference time intervention: Improving LLM truthfulness. In *Proceedings of Interspeech*, pages 4094–4098.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Chonghe Jiang, Bao Nguyen, Anthony Man-Cho So, and Viet Anh Nguyen. 2025. Probe-free low-rank activation intervention. *arXiv preprint arXiv:2502.04043*.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Junyi Li, Tianyi Tang, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. 2024a. Pre-trained language models for text generation: A survey. *ACM Computing Surveys*, 56(9):1–39.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2024b. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2021. TruthfulQA: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*.

Alisa Liu, Maarten Sap, Ximing Lu, Swabha Swayamdipta, Chandra Bhagavatula, Noah A Smith, and Yejin Choi. 2021. Dexperts: Decoding-time controlled text generation with experts and anti-experts. *arXiv preprint arXiv:2105.03023*.

Samuel Marks and Max Tegmark. 2023. The geometry of truth: Emergent linear structure in large language model representations of true/false datasets. *arXiv preprint arXiv:2310.06824*.

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.

Luca Moschella, Valentino Maiorca, Marco Fumero, Antonio Norelli, Francesco Locatello, and Emanuele Rodolà. 2023. Relative representations enable zero-shot latent space communication.

Niklas Muennighoff, Iman Dehghani, Philip Zhao, Ryo Hayashi, Genta Takuma, Carlos Muñoz Coto, Cedric Akiki, and Junyang Guo. 2024. Olmoe: Open mixture-of-experts language models. *arXiv preprint arXiv:2409.02060*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022a. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

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

Samuel J. Paech, Marina Danilevsky, Mathias Hagen, and Varun Kohli. 2023. Eq-bench: An emotional intelligence benchmark for large language models. *arXiv preprint arXiv:2312.06281*.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with GPT-4. *arXiv preprint arXiv:2304.03277*.

Luiza Pozzobon, Beyza Ermis, Patrick Lewis, and Sara Hooker. 2023. Goodtriever: Adaptive toxicity mitigation with retrieval-augmented models. *arXiv preprint arXiv:2310.07589*.

András Prékopa. 1995. *Stochastic Programming*. Springer Science & Business Media.

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

Vipula Rawte, Swagata Chakraborty, Agnibh Pathak, Anubhav Sarkar, SM Tonmoy, Aman Chadha, Amit P Sheth, and Amitava Das. 2023. The troubling emergence of hallucination in large language models–an extensive definition, quantification, and prescriptive remediations. *arXiv preprint arXiv:2310.04988*.

Shashwat Singh, Shauli Ravfogel, Jonathan Herzig, Roee Aharoni, Ryan Cotterell, and Ponnurangam Kumaraguru. 2024. Representation surgery: Theory and practice of affine steering. In *Forty-first International Conference on Machine Learning*.

Nishant Subramani, Nivedita Suresh, and Matthew E Peters. 2022. Extracting latent steering vectors from pretrained language models. *arXiv preprint arXiv:2205.05124*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models*, 3(6):7.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

SM Tonmoy, SM Zaman, Vinija Jain, Anku Rani, Vipula Rawte, Aman Chadha, and Amitava Das. 2024. A comprehensive survey of hallucination mitigation techniques in large language models. *arXiv preprint arXiv:2401.01313*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

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

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J Vazquez, Ulisse Mini, and Monte MacDiarmid. 2023. Activation addition: Steering language models without optimization. *arXiv preprint arXiv:2308.10248*.

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.

Peng Wang, Zhenyu Liu, Guoqing Liu, Xiuying Li, Zhaopeng Tu, Jie Fu, Zheng Zhang, and Juanzi Xie. 2024. Wise: Rethinking the knowledge memory for lifelong model editing of large language models. In *Advances in Neural Information Processing Systems*, volume 37, pages 53764–53797.

Song Wang, Yaochen Zhu, Haochen Liu, Zaiyi Zheng, Chen Chen, et al. 2023a. Knowledge editing for large language models: A survey. *arXiv preprint arXiv:2310.16218*.

Xiaohan Wang, Mengzhou Lin, Hongyin Gu, Meng Wu, Haotian Wang, and William Yang Wang. 2023b. Editing conceptual knowledge for large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13478–13493.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. Self-instruct: Aligning language models with self-generated instructions. *arXiv preprint arXiv:2212.10560*.

Ziwei Xu, Sanjay Jain, and Mohan Kankanhalli. 2024. Hallucination is inevitable: An innate limitation of large language models. *arXiv preprint arXiv:2401.11817*.

Zonghan Yang, Xiaoyuan Yi, Peng Li, Yang Liu, and Xing Xie. 2022. Unified detoxifying and debiasing in language generation via inference-time adaptive optimization. *arXiv preprint arXiv:2210.04492*.

Fangcong Yin, Xi Ye, and Greg Durrett. 2024. LoFiT: Localized fine-tuning on LLM representations. *arXiv preprint arXiv:2406.01563*.

Hanqing Zhang, Haolin Song, Shaoyu Li, Ming Zhou, and Dawei Song. 2023. A survey of controllable text generation using transformer-based pre-trained language models. *ACM Computing Surveys*, 56(3):1–37.

Ningyu Zhang, Yunzhi Yao, Bozhong Tian, Peng Wang, Shumin Deng, Mengru Wang, Zekun Xi, Shengyu Mao, Jintian Zhang, Yuansheng Ni, et al. 2024. A comprehensive study of knowledge editing for large language models. *arXiv preprint arXiv:2401.01286*.

A Additional Results on TruthfulQA Benchmark

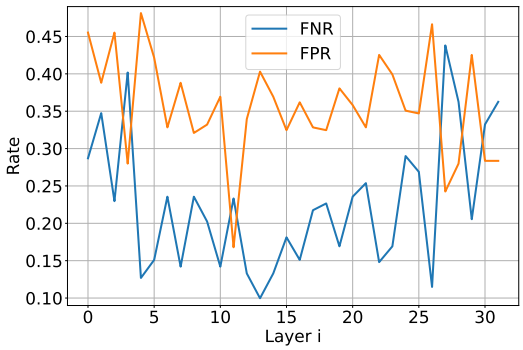
A.1 Information about TruthfulQA Dataset

The TruthfulQA dataset is a Question-Answer dataset containing 817 questions that likely elicit false answers from humans due to common misconceptions. We follow the same data-processing used in Li et al. (2024b) and Yin et al. (2024) that splits the dataset into train/validation/test with the rate of $326/82/407$ questions and utilizes two-fold cross-validation. Each question has an average length of nine words and has two sets of desirable and undesirable answers. Following Li et al. (2024b), we separate the original dataset into 5918 question-answer pairs; each has a binary label, indicating desirability. Only pairs associated with questions in the training dataset are used to create our intervention policy, while those in the validation test are set aside for parameter tuning.

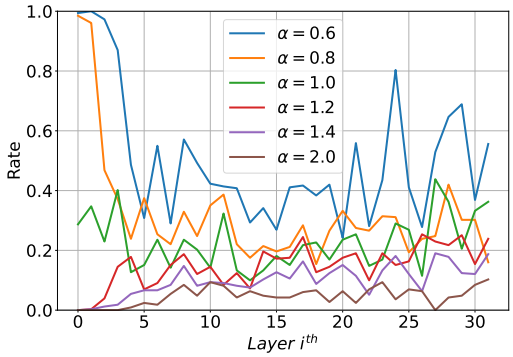
A.2 Additional Ablation Studies

A.2.1 Layer Selection Threshold with the Smooth Probing Loss

Figure 2 presents the FNR and FPR results for the layerwise probes on Llama-7B on the TruthfulQA dataset. From Figure 2a, one observes that the optimal layer tends to be a mid-layer (ℓ between 11 and 14) with smaller FNR and FPR values. Figure 2b shows that increasing α will dampen the FNR rate across layers.



(a) False Negative Rate (FNR) and False Positive Rate (FPR) across layers for intervention threshold $\tau = 11$.



(b) FNR across layers for different value of regularization parameter α of the risk-aware loss in (2).

Figure 2: FNR and FPR metrics with different hyperparameters α across layers of Llama-7B.

A.2.2 Loss Function of the Classifier

To evaluate the effectiveness of our risk-aware loss function in (2), we conduct an ablation study that compares it with weighted negative log-likelihood (NLL) loss. The weighted NLL loss to train the classifier is defined as:

$$\mathcal{L}_{\text{NLL}}(\theta_{\ell h}, \vartheta_{\ell h}) = -\sum_{i=1}^{N} [w_1 y_i^* \log(p_i) + w_0 (1 - y_i^*) \log(1 - p_i)],$$

where $p_i = \text{sigmoid}(\vartheta_{\ell h} + \theta_{\ell h}^\top a_{\ell h,i})$ is the predicted probability for sample i, and w_1 and w_0 represent the weights applied to positive and negative samples, respectively. These weights are chosen to align with the class-imbalance handling and error-prioritization approach in our proposed loss. Specifically, we set

$$w_0 = \frac{1}{N_0}, \quad w_1 = \frac{\alpha}{N_1}$$

to be consistent with the definitions of FPR and FNR in our formulation. We follow the same procedure to choose α as we do with RADIANT and run the experiment for three models: Llama-7B and Llama3-8B on the TruthfulQA dataset. The experimental setup is the same as in 4.1. The layers for intervention found by our framework with the NLL loss coincide with the layer found by our proposed loss (2) across all three models. We report the experimental results in Table 5.

Methods	**True*Info (%)** ↑	**True (%)** ↑	**MC1** ↑	**MC2** ↑
Unintervened	21.15	22.16	25.58	40.54
RADIANT	37.64	43.78	**31.98**	46.11
(NLL) RADIANT	**40.36**	**44.48**	30.91	**46.13**

Table 4: Results for Llama-7B.

Our results show that, while the weighted NLL provides a strong baseline, RADIANT consistently outperforms it in most metrics and models. In particular, RADIANT improves performance in MC1

Methods	True*Info (%) ↑	True (%) ↑	MC1 ↑	MC2 ↑
Unintervened	32.88	44.18	30.36	48.98
RADIANT (NLL)	38.36	47.25	32.12	49.83
RADIANT	**37.78**	**50.82**	**33.82**	**52.98**

Table 5: Results for Llama3-8B.

and MC2, demonstrating a stronger ability to guide model interventions. Although weighted NLL is a standard formulation, our findings suggest that RADIANT's approach to error weighting provides additional benefits beyond simple likelihood-based weighting.

We provide theoretical intuition for why the RADIANT loss might lead to superior performance in practical classification tasks compared to weighted NLL in our problem. The weighted NLL measures the classification confidence via the negative logarithm of predicted probabilities. Thus, large mispredictions incur exponentially large penalties:

$$-\log(p_i) \xrightarrow[p_i \to 0]{} \infty, \quad -\log(1-p_i) \xrightarrow[p_i \to 1]{} \infty.$$

This exponential penalty will ensure that the boundary does not go deeper into the area of any classes because if that happens, many samples will have $p_i \to 1$, then the exponential penalty will pull it back. In our problem, the false negative samples are critical, so we expect the boundary to go deeper into the desirable area, and this loss is not suitable.

Methods	Llama 7B	Llama3-8B
RADIANT(NLL)	0.104	0.128
RADIANT	0.023	0.054

Table 6: False Negative Rate comparison between RADIANT loss and Weighted NLL loss denoted as RADIANT(NLL).

Instead, RADIANT measures risk more directly (FPR, FNR), penalizing misclassification linearly in probability. Thus, while weighted NLL severely penalizes misprediction (logarithmic scale), RADIANT loss penalizes misclassification risk proportionally and is, therefore, more reasonable for our problem. Moreover, the exponential penalty potentially makes weighted NLL brittle or overly sensitive to outliers, which often exist in deep layers' activations of complex architectures like transformers. This intuition is backed up by our empirical observation that the False Negative of the weighted NLL loss is consistently higher than our loss across two models, Llama 7B and Llama3-8B.

A.2.3 The Effect of Γ and α on the Performance of RADIANT

The hyperparameter α controls the conservativeness of the classifier in terms of the False Negative Rate. High values of α ensure that no undesirable content goes undetected. However, excessively large values of α may lead to trivial classifiers that classify all samples as undesirable. Such classifiers can be identified by checking if their False Positive Rate in the validation set is one. Therefore, for a given α, along with other performance metrics, we report the average False Positive Rate and the average False Negative Rate across all trained classifiers on the validation set denoted $\overline{\text{FPR}}$ and $\overline{\text{FNR}}$.

In Table 8, we present metrics on the validation set while varying α within the set $\{1.0, 1.5, 2.0, 2.5, 3.0\}$. We use the base model Llama-7B. RADIANT's performance improves as α increases until a significant drop occurs when trivial classifiers dominate at $\alpha = 3.0$. This observation supports our approach of selecting α as high as possible without encountering the trivial classifiers issue. However, the information score decreases as α increases. This decrease can be attributed to RADIANT becoming more conservative and avoiding providing uncertain information. In practice, depending on the information sensitivity of the application of LMs, we can select α as a trade-off between the accuracy of the information and the informativeness. For example, LMs in the medical or legal sectors should avoid providing incorrect or uncertain information, so high values of α are recommended.

We report the performance metrics of Llama-7B when varying Γ in Table 7. This hyperparameter decides how much RADIANT post-intervention activations deviate from the original ones if detected as undesirable. We observe that the True score of RADIANT increases in Γ. This is because the increasing value of Γ drives activations to reside deeper inside the desirable area, thus increasing the probability of desirable generation. However, the larger value of Γ makes activations move farther from the original value, as shown by the increase in the CE and KL metrics. The extreme deviation from the original activations leads to inconsistency in semantics. It creates more non-natural

Γ	True * Info (%) ↑	True (%) ↑	Info (%) ↑	MC1 ↑	MC2 ↑	CE ↓	KL ↓
Unintervened	21.15	22.16	95.47	25.58	40.54	2.13	0.00
5	26.14	28.40	92.04	26.81	41.91	2.14	0.01
10	33.04	36.11	91.49	27.17	43.11	2.17	0.04
15	40.36	44.48	90.75	30.91	46.13	2.19	0.07
20	36.59	43.46	84.20	28.15	44.92	2.29	0.18

Table 7: The performance of RADIANT when varying Γ and fixing α of 2.5.

α	True * Info (%) ↑	True (%) ↑	Info (%) ↑	$\overline{\text{FPR}}$ ↓	$\overline{\text{FNR}}$ ↓	CE ↓	KL ↓
Unintervened	21.15	22.16	95.47	-	-	2.13	0.00
1.0	24.39	25.95	94.00	0.32	0.32	2.14	0.01
1.5	29.07	31.95	91.00	0.67	0.11	2.18	0.05
2.0	34.75	39.54	91.88	0.76	0.05	2.19	0.06
2.5	40.36	44.48	90.75	0.78	0.00	2.19	0.07
3.0	34.21	38.92	87.88	0.97	0.00	2.20	0.13

Table 8: The performance of RADIANT when varying α and fixing Γ of 15.

sentences, which can be observed at $\Gamma = 20$ with the drop in the Information score. Therefore, a reasonable score should balance the True and Information scores.

In our implementation, for each pre-trained model, we perform a grid search where α ranges over $\{1.0, 1.5, 2.0, 2.5\}$ and Γ over $\{5, 7.5, 10, 15, 20\}$ to select the optimal combination based on the True * Info score in the validation set. After running RADIANT with various pre-trained models, we find that the combination of $\Gamma = 15$ and $\alpha = 2.5$ performs effectively in most cases. Unless otherwise specified, we utilize these values for our experiments.

A.3 RADIANT Enhances Performance with Minimal Distribution Shift

Since headwise intervention applies a linear mapping to transform undesirable activations into desirable ones, it raises concerns about potential unintended shifts in meaning. To mitigate the risk of semantic drift, our intervention method is explicitly constrained via semidefinite programming to minimize the shift in activation space while enforcing a probabilistic guarantee that the modified activation crosses the classifier's decision boundary.

To showcase that the meaning shift caused by RADIANT is minimal, we report two additional metrics: Kullback-Leibler (KL) divergence of the model's next-token prediction distribution (pre- vs. post-intervention) and Cross-Entropy (CE) loss. These metrics quantify the shift in the generation distribution following the intervention. Lower values indicate smaller deviation from the original model's behavior, reducing the likelihood of unnatural outputs or anomalous characters. The calculation details are provided in Li et al. (2024b). Due to space constraints, these metrics were omitted from the main paper.

We report the KL and CE values in Table 9, which includes all the settings reported in Table 1 from the main text. Our results show that RADIANT maintains comparable KL and CE values across various scenarios, demonstrating that it preserves the original distribution while significantly improving truthfulness.

A.4 Computational Cost – Paralled Version

This section studies the impact of ITI and RADIANT on the base models' inference speed. From the theoretical aspect, it is evident that a head intervention of ITI, which is just a vector addition, is faster than that of RADIANT, which comprises a matrix multiplication and addition operator. This observation is proved again by the empirical results shown in Table 10. This table reports the average percentage increase in inference time per answer of ITI and RADIANT across the base models. It is observed that the normal version of RADIANT imposes more additional time in inference than ITI does. However, it should be noted that all RADIANT interventions are conducted on the same layer, while ITI interventions are carried out on multiple pairs of layer heads. This attribute of RADIANT allows us to parallel the interventions, which is impossible for ITI. We denote the parallel version of RADIANT as RADIANT-P and include it in Table 10. RADIANT-P offers the same decent results as RADIANT but imposes less computation cost to base models than RADIANT and ITI.

Model	Methods	CE ↓	KL ↓
Llama-7B	Unintervened	2.13	0.00
	ITI	2.20	0.07
	FSP	2.13	0.00
	NL-ITI	2.19	0.07
	LITO	2.19	0.07
	RADIANT (ours)	2.19	0.07
	FSP + ITI	2.20	0.07
	FSP + NL-ITI	2.20	0.07
	FSP + LITO	2.20	0.07
	FSP + RADIANT (ours)	2.20	0.08
Llama3-8B	Unintervened	2.38	0.00
	ITI	2.50	0.13
	FSP	2.38	0.00
	NL-ITI	2.50	0.13
	LITO	2.48	0.11
	RADIANT (ours)	2.48	0.08
	FSP + ITI	2.48	0.14
	FSP + NL-ITI	2.49	0.14
	FSP + LITO	2.54	0.17
	FSP + RADIANT (ours)	2.52	0.15
Llama2-chat-13B	Unintervened	2.31	0.00
	ITI	2.32	0.17
	FSP	2.31	0.00
	NL-ITI	2.33	0.17
	LITO	2.34	0.18
	RADIANT (ours)	2.35	0.18
	FSP + ITI	2.33	0.13
	FSP + NL-ITI	2.34	0.15
	FSP + LITO	2.36	0.17
	FSP + RADIANT (ours)	2.38	0.18
Alpaca	Base	2.81	0.00
	+ ITI	2.88	0.14
	+ RADIANT (ours)	2.81	0.13
Vicuna	Base	2.67	0.00
	+ ITI	2.77	0.26
	+ RADIANT (ours)	2.73	0.27
Llama variants + LOFiT	LOFiT (7B)	2.35	0.00
	+ ITI	2.55	0.14
	+ RADIANT (ours)	2.56	0.13
	LOFiT (8B)	3.27	0.00
	+ ITI	3.33	0.08
	+ RADIANT (ours)	3.38	0.11
	LOFiT (Chat-13B)	2.52	0.00
	+ ITI	2.73	0.21
	+ RADIANT (ours)	2.73	0.20

Table 9: Quantitative results of different intervention methods on the TruthfulQA dataset, across different Language Models and fine-tuning approaches. Parameters of RADIANT: $\alpha = 2.5, \Gamma = 15$.

A.5 Transferability Experiments

We train the intervention mappings for Llama-7B using the TruthfulQA dataset, but we evaluate its performance on the NQOpen dataset (Kwiatkowski et al., 2019). The NQOpen dataset contains approximately 3600 question-answer pairs. Our intervention vectors show strong performance on the NQOpen dataset, shown in Table 11. This effectiveness is also observed with ITI, as noted in its original paper. Nevertheless, our experiment indicates that our intervention mappings offer superior transferability and generality compared to ITI's. This experiment demonstrates RADIANT's effectiveness and highlights the generality of the computed intervention for NLP tasks.

B Additional Comparison Results

B.1 Evaluation on GPT and Gemma Base Models

In this experiment, we study the performance of finetuning-free techniques, including ITI, RADIANT, and FSP, on Gemma-2B (Team et al., 2024) and GPT-2 Large (Radford et al., 2019), which serve as alternative base models to the Llama model family. Table 12 shows that RADIANT, using few-shot prompting, outperforms other methods by a large gap. In particular, FSP + RADIANT improves the True * Info score of Gemma-2B and GPT-2 Large by 25.14% and 16.16%, respectively. Notably, FSP + RADIANT is superior to FSP + ITI in both True * Info and True and MC1 scores. Concurrently, RADIANT, implemented separately, outperforms ITI and FSP in terms of True * Info and True scores, while only slightly behind in MC1 and MC2.

B.2 Evaluation on Mistral and Qwen Base Models

RADIANT has already been evaluated on base models beyond Llama, including Gemma-2B and GPT-2 Large (Appendix B.1). Because RADIANT relies solely on activation values (rather than logits or internal weights), it is directly applicable to any transformer-based model. To further validate this, we conducted an experiment on Mistral-7B-Instruct-v0.2 and Qwen2-7B-Instruct base models and evaluated the results using GPT4. The optimal intervention layers found by RADIANT are 18 and 25, respectively, for Qwen2-7B-Instruct and Mistral-7B-Instruct-v0.2.

Tables 13 and 14 show that RADIANT improves truthfulness and overall performance across different models. For Mistral-7B-Instruct-v0.2, RADIANT achieves the highest True * Info (81.94%) and True (68.17%) scores, outperforming both the baseline and ITI. For Qwen2-7B-Instruct, RADIANT leads in True * Info (77.51%) and True (53.05%).

Base models	ITI	RADIANT	RADIANT-P
Gemma-2B	2.53	6.82	1.75
GPT-2 Large	2.43	3.01	1.65
Llama-7B	2.46	3.09	1.45
Llama3-8B	2.51	3.32	1.55
Llama2-chat-13B	2.51	4.72	1.57

Table 10: The average percentage increase in inference time per answer of ITI and RADIANT across base models.

Dataset	Methods	True * Info (%) ↑	True (%) ↑	MC1 ↑	MC2 ↑	CE ↓	KL ↓
NQOpen	Unintervened	17.16	18.50	40.90	53.10	2.13	0.00
	ITI	16.97	18.90	40.40	52.94	2.20	0.07
	RADIANT (ours)	**20.66**	**22.10**	**41.50**	**54.38**	2.16	0.04
TriviaQA	Unintervened	87.82	92.25	32.60	64.35	2.13	0.00
	ITI	91.14	94.20	32.70	65.16	2.21	0.09
	RADIANT (ours)	**92.35**	**96.50**	**35.30**	**67.20**	2.23	0.09

Table 11: Quantitative results of the transferability of RADIANT's intervention on different datasets.

Regarding comparisons with knowledge editing methods such as WISE (Wang et al., 2024) and ConceptEdit (Wang et al., 2023b), we emphasize that our work targets a different setting: inference-time intervention rather than parameter-based model editing. While model editing methods aim to alter stored knowledge via weight updates, RADIANT operates at inference time by activation intervention, enabling lightweight, reversible control over generation.

B.3 Evaluation on Sparse MoE Architectures

We show that RADIANT can be applied to a sparse Mixture of Experts. The core mechanism of RADIANT, layerwise and headwise activation editing, relies solely on access to activations at a given layer and does not depend on weight sharing or architectural uniformity. Moreover, since our method operates directly on attention head activations, and most sparse MoE designs apply sparsity to MLP layers rather than attention blocks, the architecture-level sparsity introduced by MoEs does not interfere with our intervention method. To verify our performance on this MoE architecture, we conduct experiments with a sparse MoE model, OLMoE-1B-7B-0924, from Muennighoff et al. (2024). We provide the results of our method, ITI, and base model for the TruthfulQA dataset as follows.

Table 15 shows that RADIANT works well for the OLMoE-1B-7B-0924 model, boosting the True score by nearly 8%. RADIANT outperforms ITI regarding True * Info, True, MC1, and MC2. This showcases its effectiveness in generating truthful answers in architectures like Sparse MoEs.

B.4 Comparison with Supervised Fine-Tuning

Supervised fine-tuning (SFT) attempts to align LLMs with human preferences (Ouyang et al., 2022b). Given a prompt, SFT encourages the model to generate desirable answers and reduce the likelihood of generating undesirable answers by optimizing the cross-entropy loss. However, SFT's requirement to fine-tune all LLM parameters demands substantial GPU resources for the back-propagation operations. Due to computational constraints, we can only perform SFT on the GPT2-large, the smallest model in our experiments.

Table 12 highlights the advantages of inference-time methods like ours: by avoiding gradient computation or backpropagation, they offer a lightweight, fast, versatile, and economical way to improve the performance of LLMs. This is especially useful in low-resource scenarios. Because Llama-7B is used as a base model for many of our experiments, we also include the results of SFT on Llama-7B for comparison, but it is worth noting that the number is taken from the ITI paper (Li et al., 2024b). Since our evaluation framework differs from ITI in terms of the GPT-judge and GPT-info models, which is attributed to the fact that these models in the ITI paper are no longer available on OpenAI, the results may not be fair for comparison. From Table B.1, SFT achieves the best performance in terms of MC metrics and reaches a high score of True * Info and True. Regarding the True score, RADIANT still outperforms SFT in the individual and integrating versions with FSP, offering 38.73% and 40.41% correct answers, respectively. When combined with FSP, RADIANT achieves 35.36% in True * Info score, surpassing

Methods	True * Info (%) ↑	True (%) ↑	MC1 ↑	MC2 ↑	CE ↓	KL ↓
Unintervened	31.00	51.23	27.12	43.62	2.55	0.00
ITI	33.42	54.74	29.14	46.01	2.64	0.17
FSP	34.92	42.23	**35.10**	**49.24**	2.55	0.0
RADIANT(ours)	**35.62**	**59.62**	30.34	48.06	2.62	0.15
FSP + ITI	48.83	61.57	38.27	54.73	2.69	0.16
FSP + RADIANT(ours)	**56.14**	**64.71**	**39.54**	**56.98**	2.65	0.09

(a) Gemma-2B

Methods	True * Info (%) ↑	True (%) ↑	MC1 ↑	MC2 ↑	CE ↓	KL ↓
Unintervened	19.20	21.91	23.57	40.75	2.8	0.0
SFT	**35.16**	38.28	**35.70**	**53.57**	3.27	0.46
ITI	26.94	31.09	24.68	42.31	2.94	0.13
FSP	21.82	27.30	25.34	42.07	2.8	0.0
RADIANT (ours)	30.18	**38.73**	25.14	42.14	2.92	0.12
FSP + ITI	29.53	30.45	25.12	**44.79**	2.98	0.18
FSP + RADIANT (ours)	**35.36**	**40.41**	**26.18**	44.29	2.94	0.16

(b) GPT-2 Large

Table 12: Quantitative results of different intervention methods on TruthfulQA dataset, across different language models. Parameters of RADIANT: $\alpha = 2.5, \Gamma = 15$.

Methods	**True * Info (%)** ↑	**True (%)** ↑	**MC1** ↑	**MC2** ↑
Unintervened	80.05	67.20	**55.69**	70.93
ITI	76.87	62.30	54.23	70.41
RADIANT (ours)	**81.94**	**68.17**	55.23	**71.75**

Table 13: Experimental results of baselines for Mistral-7B-Instruct-v0.2.

SFT but requiring fewer resources. For the implementation of SFT, we use the SFTTrainer framework[2], one of the most popular frameworks for this algorithm. While we remained almost the default parameters proposed by the library, we had to tune many important parameters like learning rate, parameters of the Adam optimizer, weight decay, and so on, to get a consistent and stable fine-tuned model. Some important parameters for SFT are reported in Table 16, while its best performance is shown previously in Table 12. This observation strongly supports the practicability of RADIANT, which only necessitates tuning two key hyperparameters α in the probe loss (2), and $\Gamma = \Phi^{-1}(1-\gamma)$ in the computation of the intervention map (4). A detailed analysis of these parameters to provide insight into their impact is presented in Appendix A.2.3. This section offers useful insights and guidelines for selecting values for any new models. Furthermore, compared to other methods like ITI, the grid search on two hyperparameters like ours is efficient and reasonable, so it is not harder to tune the hyperparameters of RADIANT than other previous works.

[2] https://huggingface.co/docs/trl/en/sft_trainer

C Experiments on Other NLP Tasks

C.1 Toxicity Mitigation Benchmark

In this section, we show the performance of RADIANT in mitigating toxicity in long-form text generation. In this task, the language models are required to complete an incomplete prefix piece of a text. Normally, the prefix prompt is selected to elicit toxic content from LLMs. For a fair comparison to previous works, we set up experiments following Singh et al. (2024) and Pozzobon et al. (2023), which is detailed below.

Training dataset. We use the Toxic Comments Classification Challenge data.[3] The dataset comprises sentences and their human toxicity labels. We follow the data preprocessing steps from Singh et al. (2024) while the activations gathering is identical to the procedure of the QA task.

Models. Following existing works in the field, we adopt the GPT2-Large as the base model across all experiments of the toxicity mitigation task.

Hyperparameter As mentioned in the QA task section, there are two important hyperparameters in our framework, namely α, and $\Gamma = \Phi^{-1}(1-\gamma)$,

[3] https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge

Methods	True * Info (%) ↑	True (%) ↑	MC1 ↑	MC2 ↑
Unintervened	64.13	50.55	40.02	59.89
ITI	73.68	50.30	**40.88**	60.86
RADIANT (ours)	**77.51**	**53.05**	40.39	**61.17**

Table 14: Experimental results of baselines for Qwen2-7B-Instruct.

Methods	True * Info (%) ↑	True (%) ↑	MC1 ↑	MC2 ↑	CE ↓	KL ↓
Unintervened	23.43	31.42	28.92	45.42	2.67	0.00
ITI	25.52	32.74	28.48	45.50	2.84	0.13
RADIANT (ours)	**28.56**	**39.17**	**30.13**	**47.05**	2.68	0.12

Table 15: Performance on TruthfulQA with OLMoE-1B-7B-0924.

Parameter	Value
learning_rate	0.00002
weight_decay	0
adam_beta1	0.8
adam_beta2	0.999
adam_epsilon	1×10^{-8}
max_grad_norm	1
batch_size	32
epochs_num	5
lr_scheduler_type	linear

Table 16: Parameter values for SFT.

which would be selected by a grid search procedure detailed in Appendix A.2.3.

Baselines. We include several baselines that have the same goal of reducing the toxicity of LLMs, including MIMIC (Singh et al., 2024), DEXPERTS (Liu et al., 2021), DAPT (Gururangan et al., 2020), UDDIA (Yang et al., 2022), PPLM (Dathathri et al., 2019), GOODTRIEVER (Pozzobon et al., 2023). As for MIMIC, we consider two versions: Mean Matching (MM) and Mean+Covariance Matching (MCM), introduced in their original paper.

Metrics. We assess the performance using three key metrics: toxicity, fluency, and diversity.

Hyperparameter	Value
Number of Samples	25
Max Length	20
Temperature	1
Top-p (sampling)	0.9
Top-k (sampling)	0

Table 17: Hyperparameter settings for the decoding mechanism in the toxicity mitigation task

(i) Toxicity: we use the non-toxic split of RealToxicityPrompts (Gehman et al., 2020) and utilize the evaluation framework in Liu et al. (2021) and Singh et al. (2024). For each prompt in the dataset, the models generate 25 outputs, each capped at 20 tokens in length. The parameters of the shared decoding mechanism of all algorithms are presented in Table 17. These outputs are analyzed using Perspective API,[4] which estimates the likelihood that a human would perceive the text as toxic. Two metrics are derived:

- Expected Maximum Toxicity is denoted as Exp. Max. Tox. We identify the output with the highest toxicity score for every prompt and compute the average of these maximum scores across all prompts.
- Toxic Completion Proportion is abbreviated as Tox. Prob. This metric tracks the fraction of outputs considered toxic, where toxicity is defined as a score above 0.5 based on the Perspective API's threshold.

(ii) Fluency is evaluated by calculating the perplexity of the generated outputs, using GPT-2 (XL) as a reference model. Lower perplexity values suggest that the text is more coherent and grammatically fluent.

(iii) Diversity is assessed by examining the ratio of unique n-grams (1-gram, 2-gram, and 3-gram) to the total number of tokens in the generated text. This metric captures the range of variation in the outputs, with higher values indicating more diverse and varied language use. This methodology ensures a balanced evaluation, providing insights into the ability of models for non-toxic, fluent, and diverse generation.

[4] https://perspectiveapi.com/

Results. The experimental results of the baselines are shown in Table 18, where the base model for all methods is GPT-2 Large. The result of the original model is described in the first row. We divide the baselines into two groups. Using an extensive fine-tuning procedure, the first group comprises DAPT, GeDI, PPLM, UDDIA, DExperts, and GOODTRIEVER. In contrast, the second group contains inference-time fine-tuning-free methods like MIMIC, ITI, and RADIANT. The baselines in the first group are better than their counterparts in the second group regarding toxicity metrics. However, these methods require fine-tuning or computing gradients at inference time, which can be computationally intensive. MIMIC, ITI, and RADIANT achieved a toxicity reduction comparable to many algorithms in the first group, but consumed much fewer resources. Specifically, RADIANT is superior to PPLM and is equally competitive to DAPT. In particular, RADIANT offers the best toxicity reduction impact within the second group compared to ITI and MIMIC while maintaining a better fluency and diversity of generated sentences. The fluency of RADIANT is even more favored than almost all algorithms in the first group, except for UDDIA. At the same time, its diversity metric is better than that of other baselines except for PPLM.

C.2 Long-Fact Generation Benchmark

We inspected the performance of our method and ITI on the Long-Fact task. We followed the experimental setup and prompt format in Cheng et al. (2024) and used their code repository[5] for evaluation. This task requires models to generate detailed document-length descriptions of queried objects, often exceeding 1,000 tokens. Evaluation involves breaking responses into atomic facts using LLaMA3.1-70B-Instruct and evaluating their truthfulness with GPT-4. Metrics include Precision (truthful fact proportion), Recall@128 (truthful facts per 128), and F1@128. Results are based on 120 samples. Because we do not have a set of true and wrong samples in this dataset, we cannot learn the mapping for ITI and RADIANT. Therefore, we transfer the learned mappings of ITI and RADIANT from Truthful QA for Qwen2-7B-Instruct and Mistral-7B-Instruct-v0.2.

The results for Qwen2-7B-Instruct are reported in Table 20, while the results for Mistral-7B-Instruct-v0.2 are reported in Table 21.

The results demonstrate that RADIANT's intervention mapping effectively enhances truthfulness in the long-fact task, showcasing strong generalization across models. In contrast, ITI's mapping transfers well to Qwen2-7B-Instruct but performs poorly on Mistral-7B-Instruct-v0.2, highlighting its limited adaptability. RADIANT achieves the highest precision across both models, indicating its strength in generating truthful facts. While ITI slightly outperforms in recall for Qwen2-7B, and the unintervened model leads in recall for Mistral-7B, RADIANT maintains a strong balance between precision and recall. This is reflected in its highest F1@128 scores, making it the most effective overall. The small variations in recall suggest that interventions refine the accuracy of the facts rather than significantly increasing the number of truthful facts generated.

C.3 Creative Writing Benchmark

To further address the concern about meaning shift, we tested our method on the Creative Writing v3 task from the EQ-Bench benchmark (Paech et al., 2023) using Qwen2-7B-Instruct. This experiment evaluates how intervention mapping affects the meaning of creative text. The ELO scores in Table 19 show a slight decrease when applying ITI and RADIANT, which is expected, since these interventions enhance truthfulness, potentially limiting word choice flexibility. However, the results indicate that the interventions do not significantly harm the model's creative capabilities.

D Mathematical Proof

Proof of Theorem 1. The logistic classifier $\mathcal{C}_{\ell h}$ output a prediction 0 if $\vartheta_{\ell h} + \theta_{\ell h}^\top a_{\ell h} < 0$. If $\mathbb{Q}_{\ell h}$ is Gaussian $\mathcal{N}(\mu, \Sigma)$, then by (Prékopa, 1995, Theorem 10.4.1), the probability constraint of (3) can be written as

$$\vartheta_{\ell h} + \theta_{\ell h}^\top \mu + \Phi^{-1}(1-\gamma)\sqrt{\theta_{\ell h}^\top \Sigma \theta_{\ell h}} \leq 0.$$

Next, we add an auxiliary variable $t \in \mathbb{R}_+$ with an epigraph constraint $\sqrt{\theta_{\ell h}^\top \Sigma \theta_{\ell h}} \leq t$. Because $\Phi^{-1}(1-\gamma) > 0$ for $\gamma \in (0, 0.5)$, problem (3) is equivalent to

$$\begin{aligned} \min \quad & \|\mu - \widehat{\mu}\|_2^2 + \|\Sigma^{\frac{1}{2}} - \widehat{\Sigma}^{\frac{1}{2}}\|_F^2 \\ \text{s.t.} \quad & \vartheta_{\ell h} + \theta_{\ell h}^\top \mu + \Phi^{-1}(1-\gamma)t \leq 0, \\ & \sqrt{\theta_{\ell h}^\top \Sigma \theta_{\ell h}} \leq t \\ & \mu \in \mathbb{R}^d,\ \Sigma \in \mathbb{S}_+^d,\ t \in \mathbb{R}_+. \end{aligned}$$

[5] https://github.com/YiCheng98/IntegrativeDecoding

Model	Exp. Max. Tox. ↓	Tox. Prob. ↓	Fluency ↓	1-gram ↑	2-gram ↑	3-gram ↑
GPT-2 (large)	0.39	0.25	24.66	0.58	0.85	0.85
DAPT	0.27	0.09	30.27	0.57	0.84	0.84
GeDI	0.24	0.06	48.12	**0.62**	0.84	0.83
PPLM (10%)	0.38	0.24	32.58	0.58	**0.86**	**0.86**
UDDIA	0.24	0.04	**26.83**	0.51	0.80	0.83
DExperts	**0.21**	**0.02**	27.15	0.56	0.84	0.84
GOODTRIEVER	0.22	0.04	27.11	0.58	0.82	0.83
MM (MIMIC)	0.33	0.16	28.00	**0.58**	**0.85**	**0.85**
MCM (MIMIC)	0.29	**0.09**	30.70	0.54	0.84	0.84
ITI	0.31	0.12	33.12	0.57	**0.85**	**0.85**
RADIANT	**0.27**	**0.09**	**27.10**	**0.58**	**0.85**	**0.85**

Table 18: Quantitative results of different intervention methods on RealToxicityPrompts dataset. Parameters of RADIANT: $\alpha = 2.5, \Gamma = 15$.

Methods	**ELO ↑**
Unintervened	592.5
ITI	554.6
RADIANT (ours)	**578.6**

Table 19: ELO Scores on Creative Writing v3 Task.

Let $S \leftarrow \Sigma^{\frac{1}{2}} \in \mathbb{S}_+^d$, the constraint $\sqrt{\theta_{\ell h}^\top \Sigma \theta_{\ell h}} \leq t$ is equivalent to $\|S\theta_{\ell h}\|_2 \leq t$, which leads to (4). Thus, the optimal pushforward $\Delta_{\ell h}$ should push $\widehat{\mathbb{P}}_{\ell h} \sim \mathcal{N}(\widehat{\mu}, \widehat{\Sigma})$ to $\mathbb{Q}_{\ell h} \sim \mathcal{N}(\mu^\star, (S^\star)^2)$. One can verify through simple linear algebraic calculations that the mapping $\Delta_{\ell h}(a_{\ell h}) = G_{\ell h}^\star a_{\ell h} + g_{\ell h}^\star$ defined in the theorem statement is the desired mapping. This completes the proof. □

Methods	Precision ↑	Recall@128 ↑	F1@128 ↑
Unintervened	87.54	55.42	69.02
ITI	88.01	**56.73**	68.50
RADIANT (ours)	**88.52**	56.17	**70.13**

Table 20: Performance on the Long-Fact task with Qwen2-7B-Instruct.

Methods	Precision ↑	Recall@128 ↑	F1@128 ↑
Unintervened	88.18	**59.23**	72.15
ITI	87.82	58.73	72.20
RADIANT (ours)	**89.03**	59.22	**73.73**

Table 21: Performance on the Long-Fact task with Mistral-7B-Instruct-v0.2.