Ethos: Rectifying Language Models in Orthogonal Parameter Space

Lei Gao *, **Yue Niu** *, **Tingting Tang, Salman Avestimehr, Murali Annavaram**
University of Southern California
{leig, yueniu, tangting, avestime, annavara}@usc.edu

Abstract

Language models (LMs) have greatly propelled the research on natural language processing. However, LMs also raise concerns regarding the generation of biased or toxic content and the potential disclosure of private information from the training dataset. In this work, we present a new efficient approach, Ethos, that rectifies LMs to mitigate toxicity and bias in outputs and avoid privacy leakage. Ethos is built on task arithmetic. However, unlike current task arithmetic algorithms, Ethos distinguishes general beneficial and undesired knowledge when reconstructing task vectors. Specifically, Ethos first obtains a set of principal components from the pre-trained models using singular value decomposition. Then, by projecting the task vector onto principal components, Ethos separates the principal components that encode general from those associated with undesired knowledge. Ethos performs forgetting or unlearning by only negating the task vector with undesired knowledge, thereby minimizing collateral damage on general model utility. We demonstrate the efficacy of our approach on three different tasks: bias, toxicity, and memorization unlearning. Evaluations show Ethos is more effective in removing undesired knowledge while maintaining the overall model performance compared to current task arithmetic methods.

1 Introduction

The advent of language models (LMs) has enhanced the current capabilities in text understanding and generation (Vaswani et al., 2017; Brown et al., 2020; Touvron et al., 2023; Zhao et al., 2023). Due to their significant potential LMs have been the driving force in many automated systems that improve productivity in real-world tasks (OpenAI, 2023; Chen et al., 2021; Thoppilan et al., 2022). However, despite their success, LMs also bring to the forefront some new challenges. This paper focuses on one pivotal challenge among these: LMs' propensity to generate toxic, biased content or reveal private training records.

Overview of Toxicity/Bias/Privacy Concerns of LMs: Since LMs are pre-trained with a large volume of data, the composition of the dataset during pre-training can greatly affect the performance of LMs. In particular, suppose a dataset used in pre-training contains a substantial amount of toxic information, it can result in an LM that is likely to generate toxic or harmful messages for certain prompts (Röttger et al., 2020; Hartvigsen et al., 2022). Similarly, an imbalanced dataset with unevenly distributed data points among groups (e.g., gender, race, ethnicity) can lead to the development of biases in LMs (Bolukbasi et al., 2016; Dixon et al., 2018; Sheng et al., 2019; Gallegos et al., 2023). For instance, LMs may associate certain features with a gender group when pre-trained on gender-imbalanced datasets. Another critical concern in deploying LMs is the risk of privacy leakage due to *model memorization*. Specifically, LMs tend to overfit training data and memorize specific examples, increasing vulnerability to privacy breaches, such as training data extraction attacks (Carlini et al., 2020, 2022; Hu et al., 2021; Flemings et al., 2024). Memorization compromises privacy and poses security risks, especially when the training data contains sensitive information.

Addressing these challenges is crucial in the development of LMs. A naive approach is to retrain the model from scratch, for instance, whenever bias or memorization is discovered and removed from the training data. Considering the prohibitive costs of training LMs, it is infeasible to re-train the model. Hence, the objective of this work is to rectify LMs without incurring substantial costs.

Overview of Model Editing by Task Arithmetic. Prior work (Ilharco et al., 2023) introduces a *model editing* method that reduces toxic informa-

*These authors contributed equally.

Findings of the Association for Computational Linguistics: NAACL 2024, pages 2054–2068
June 16-21, 2024 ©2024 Association for Computational Linguistics

tion in outputs by directly editing models with a *task vector*. The task vector, obtained after fine-tuning the model on a downstream dataset, encodes certain undesired knowledge (e.g., toxicity). Therefore, negating such a task vector helps rectify LMs and forgetting or unlearning undesired bias while maintaining reasonable model performance. To further improve the model editing performance, Zhang et al. leverage parameter-efficient fine-tuning methods such as Low-Rank Adaptation (LoRA) (Hu et al., 2022) to edit the task vector formed by a subset of the model weights using parameter-efficient modules only rather than the full model weights.

Current model editing methods still struggle to maintain LMs' performance when directly operating in the parameter space. The reason is that task vectors mix undesired knowledge with the general knowledge that is necessary for preserving model utility (Hu et al., 2023). As a result, simply negating the task vector on an LM inevitably removes the general knowledge alongside the undesired knowledge, causing collateral damage to the overall model performance. We present more detailed related work in Appendix A.

Overview of the Proposed Method. To address the limitations in current model editing methods for forgetting or unlearning undesired information, we propose `Ethos`, a new model editing method that generates task vectors containing undesired knowledge only and minimizes adverse effects on LMs' performance. The core idea of `Ethos` is to analyze a model's weights in an *orthogonal space* and distinguish the components related to general knowledge from the ones associated with undesired knowledge. We first define an orthogonal parameter space with a set of orthogonal components. Specifically, we apply singular value decomposition (SVD) to the pre-trained weights and obtain the principal components. The obtained principal components serve as the bases that fully represent the weight space of the pre-trained LM.

Given the orthogonality of the principal components, we treat each as a separable component encoding specific *orthogonal knowledge*. The LM's output represents a combination of knowledge from all principal components. To identify the components for undesired knowledge, we fine-tune the pre-trained LM on a downstream task, such as a toxic dataset, and obtain an initial task vector. Then, we project the task vector onto the defined orthogonal space. The principal components that present significant changes after the projection are classified as components encoding undesired knowledge, while others with marginal changes after the projection are classified as components for general knowledge. We use all components for undesired knowledge to construct a new task vector, which is then subtracted from the pre-trained weights to mitigate toxicity, bias, or memorization in the LM.

We conduct experiments on three different tasks: bias, toxicity and memorization unlearning in LMs. We use pre-trained LMs, including OPT (Zhang et al., 2022), GPT2 (Radford et al., 2019), GPT-Neo (Black et al., 2021), and large LMs like Llama2 (Touvron et al., 2023). Evaluations show that `Ethos` effectively reduces bias, toxicity, and privacy leakage in pre-trained LMs. Notably, our approach demonstrates better unlearning performance than current model editing methods while maintaining model utility comparable to that of pre-trained models. We also conduct ablation studies to analyze various components of our methods.

2 Preliminary

2.1 Parameter-Efficient Fine-Tuning

To enhance the efficiency of fine-tuning LMs while reducing memory and computational overhead, Parameter-efficient fine-tuning (PEFT) methods have been proposed to fine-tune only a subset of the existing model parameters (Zaken et al., 2022; Houlsby et al., 2019; Li and Liang, 2021). Among these, the low-rank adaptation algorithm, LoRA (Hu et al., 2022), stands out for achieving performance comparable to full-parameter fine-tuning. For a linear layer, it freezes the pre-trained weights $W_0 \in \mathbb{R}^{d\times k}$ and injects trainable low-rank matrices $A \in \mathbb{R}^{r\times k}$ and $B \in \mathbb{R}^{d\times r}$, constraining the weight updates in a low-rank space. The total number of trainable parameters is significantly reduced given rank $r \ll min(d, k)$. The forward pass is then modified as

$$\boldsymbol{h} = W_0 \cdot \boldsymbol{x} + BA \cdot \boldsymbol{x}, \tag{1}$$

where input $\boldsymbol{x} \in \mathbb{R}^k$ and output $\boldsymbol{h} \in \mathbb{R}^d$. The matrix A is initialized from a random Gaussian distribution, and B is initialized to zero. Therefore, the output h remains the same as the original layer at the beginning of training. In this work, we use LoRA fine-tuning instead of full model fine-tuning across all experiments and use LoRA parameters A and B to construct task vectors.

2.2 Task Arithmetic

Recent advancements in model editing techniques (Cao et al., 2021; Mitchell et al., 2021, 2022; Meng et al., 2022) have seen the emergence of task arithmetic as a cost-effective and scalable method (Ilharco et al., 2023; Zhang et al., 2023a; Ortiz-Jimenez et al., 2023; Tang et al., 2023). Task arithmetic is to modify a pre-trained model directly using a vector called *task vector*. A task vector is usually attained after fine-tuning the pre-trained model on a downstream task. Specifically, given weights of a pre-trained model $\boldsymbol{\theta}_{\text{pt}}$, $\boldsymbol{\theta}_{\text{ft}}$ denotes weights after fine-tuning on a downstream task, a task vector is calculated as

$$\Delta\boldsymbol{\theta} = \boldsymbol{\theta}_{\text{ft}} - \boldsymbol{\theta}_{\text{pt}}. \tag{2}$$

As neural networks implicitly memorize knowledge in their parameters (Cao et al., 2021), the task vector obtained in Eq (2) also encodes knowledge about the downstream task. In this work, we mainly focus on the *negation* operation of the task vector defined as

$$\boldsymbol{\theta}^{*}_{\text{pt}} = \boldsymbol{\theta}_{\text{pt}} - \lambda \cdot \Delta\boldsymbol{\theta}, \tag{3}$$

where λ denotes a scaling factor that controls the weight of the task vector. Negation aims to remove specific knowledge from the pre-trained model. For instance, if a pre-trained model gives toxic or biased information, negating a task vector attained from a toxic or biased dataset can rectify the pre-trained model without incurring costly procedures such as re-training the model.

3 Methodology

The objective of this study is to edit LMs to remove certain types of undesired knowledge encoded in LMs, such as bias, toxicity, or certain private information. Existing methods that rely on task vectors are unable to distinguish undesired knowledge from overall beneficial knowledge within those vectors. Therefore, we propose Ethos that is aimed to remove only undesired knowledge and ensure the edited model is rectified without significantly compromising the model utility.

Next, we present our method, Ethos. At a high level, Ethos decomposes weights of a pre-trained model along orthogonal directions and analyzes changes in each direction when fine-tuning the pre-trained model on a downstream task. We demonstrate that each direction represents a specific type

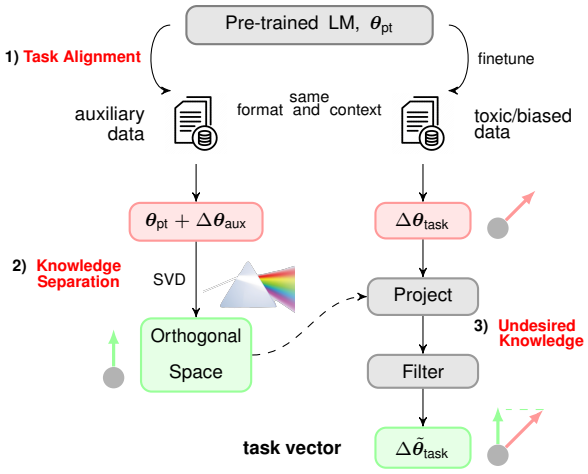

Figure 1: Overview of Ethos. Ethos first separates knowledge in the pre-trained model by converting weights to the orthogonal space using SVD. Then, Ethos projects the initial task vector, $\Delta\boldsymbol{\theta}_{\text{task}}$, to the orthogonal space, and identifies components for general knowledge and components for task-specific knowledge. At last, Ethos creates a new task vector, $\Delta\tilde{\boldsymbol{\theta}}_{\text{task}}$, with only task-specific components.

of knowledge that is orthogonal to the others. During fine-tuning, directions with general knowledge that exist in the pre-trained model will observe marginal changes, while substantial changes can happen along directions with task-specific knowledge. Therefore, Ethos constructs a new task vector only along these task-specific directions and negates the task vector on the pre-trained model. Hence, with a proper downstream dataset, one can identify orthogonal spaces that are most impacted by bias or toxic information.

As shown in Figure 1, Ethos consists of the following key steps.

Task Alignment. Given a pre-trained model, $\boldsymbol{\theta}_{\text{pt}}$, we first align it with the downstream task. Since the pre-trained model lacks knowledge about the downstream task, the alignment step is necessary for constructing an orthogonal space that captures the downstream context. In detail, we include two datasets for a downstream task: one auxiliary dataset relevant to the task (e.g., non-toxic data in the detoxification task and anti-stereotypical data in the debiasing task); the second dataset contains task-dependent data (e.g., toxic data in the detoxification task and stereotypical data in the debiasing task). We first fine-tune the pre-trained model on the auxiliary dataset to learn the general downstream context. We denote the fine-tuned model as $\boldsymbol{\theta}'_{\text{pt}} = \boldsymbol{\theta}_{\text{pt}} + \Delta\boldsymbol{\theta}_{\text{aux}}$.

Knowledge Separation. As stated in prior

works (Meng et al., 2022), $\boldsymbol{\theta}'_{\text{pt}}$ implicitly memorizes knowledge from training datasets, including general and undesired knowledge. The key first step in our method is to construct a separable space so that we can project weights onto separable directions and analyze the role of each direction.

Inspired by orthogonality in linear algebra and its applications in machine learning (Niu et al., 2023a,b, 2022), we say W_1 and W_2 encode *orthogonal knowledge* if $W_1^* \cdot W_2 = \mathbf{0}$. We can understand the definition via a linear layer in LMs. For a linear layer, given input $\boldsymbol{x}$, output after W_1 and W_2 is

$$\boldsymbol{y}_1 = W_1 \cdot \boldsymbol{x}, \qquad \boldsymbol{y}_2 = W_2 \cdot \boldsymbol{x}.$$

We can see that if W_1 is orthogonal to W_2, their outputs are also orthogonal. Specifically, $\langle \boldsymbol{y}_1, \boldsymbol{y}_2 \rangle = \boldsymbol{x}^* \cdot W_1^* W_2 \cdot \boldsymbol{x} = 0$. Therefore, given input $\boldsymbol{x}$, outputs after W_1 and W_2 contain information that is orthogonal.

With the observation above, we can convert the pre-trained model, $\boldsymbol{\theta}'_{\text{pt}}$, into an orthogonal space, where each direction can denote knowledge that is orthogonal to other directions. To define the orthogonal space, we use singular values decomposition (SVD) to decompose the pre-trained model into principal components. Given weights in i-th layer, $W \in \mathbb{R}^{n \times n}$, we decompose it as

$$W \equiv \sum_{k=1}^{n} W_k \equiv \sum_{k=1}^{n} s_k \cdot \boldsymbol{u}_k \cdot \boldsymbol{v}_k^*, \tag{4}$$

where $\boldsymbol{u}_k \cdot \boldsymbol{v}_k^*$ denote k-th principal component in W^i, s_k is k-th singular value. As each principal component W_k is orthogonal to all others, the output after W_k also represents orthogonal information to outputs from other principal components. Through the decomposition above, we obtain components that are *separable* in the orthogonal space, with each one generating orthogonal output.

Undesired Knowledge. With separable components from a pre-trained model, $\boldsymbol{\theta}'_{\text{pt}}$, if we can separate the components that represent general knowledge from undesired knowledge, model debiasing or detoxication can be effectively done by only removing those components for bias.

To that end, we fine-tune the pre-trained model, $\boldsymbol{\theta}_{\text{pt}}$, on a dataset with undesired knowledge and obtain an initial task vector, $\Delta\boldsymbol{\theta}_{\text{task}}$, as shown in Figure 1. Usually, $\Delta\boldsymbol{\theta}_{\text{task}}$ encodes both general and task-specific knowledge. We then project i-th layer's weight in $\Delta\boldsymbol{\theta}_{\text{task}}$ onto the orthogonal space of $\boldsymbol{\theta}'_{\text{pt}}$ as

$$S_{\text{task}} = U^* \cdot \Delta W \cdot V, \tag{5}$$

where $U = [\boldsymbol{u}_1, \cdots, \boldsymbol{u}_n]$, $V = [\boldsymbol{v}_1, \cdots, \boldsymbol{v}_n]$ obtained via SVD on $\boldsymbol{\theta}'_{\text{pt}}$. Each value in S_{task} denotes the singular value for the corresponding components.

We first make the following arguments:

1. If a principal component in ΔW represents general knowledge, the singular value after projection tends to be small. Since the pre-trained model comes with sufficient general knowledge, any further fine-tuning with similar knowledge will not result in substantial changes.

2. If a principal component in ΔW represents undesired knowledge, the singular value after projection tends to be large. The reason is that fine-tuning the pre-trained model on an unseen downstream task will lead to significant weight changes.

Note that since $\Delta\boldsymbol{\theta}_{\text{task}}$ and $\boldsymbol{\theta}'_{\text{pt}}$ do not share the exact principal components. The resulting S_{task} can contain non-diagonal values after the project. As a result, with the process above, we may find additional components not in $\boldsymbol{\theta}'_{\text{pt}}$. Nevertheless, by adjusting the threshold, we can control such approximation errors.

Therefore, by observing the magnitude of singular values in S_{task}, we conjecture that components with large singular values represent task-specific knowledge while components with small singular values represent general knowledge. We then construct a new task vector, $\Delta\tilde{\boldsymbol{\theta}}_{\text{task}}$, by only using components with large singular values as

$$\Delta\tilde{\boldsymbol{\theta}}_{\text{task}} = U \cdot \tilde{S}_{\text{task}} \cdot V^*, \tag{6}$$

where $\tilde{S}_{\text{task}}$ denotes the chosen large singular values. In this paper, we obtain $\tilde{S}_{\text{task}}$ as

$$\tilde{S}_{\text{task}}(i) = \begin{cases} S_{\text{task}}(i) & |S_{\text{task}}(i)| \geq \xi \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

where ξ is a threshold to define general and undesired knowledge (*Filter* in Figure 1).

Then, we perform model detoxication or debiasing as

$$\boldsymbol{\theta}^*_{\text{pt}} = \boldsymbol{\theta}_{\text{pt}} + \Delta\boldsymbol{\theta}_{\text{aux}} - \lambda \cdot \Delta\tilde{\boldsymbol{\theta}}_{\text{task}} \tag{8}$$

Figure 1 shows the overall procedure for obtaining a task vector, $\Delta\tilde{\boldsymbol{\theta}}_{\text{task}}$. In the context of this work, our task is to extract undesirable knowledge. Hence, the fine-tuning task will use datasets that

contain undesirable information, such as toxicity or bias, and find the corresponding task vector.

Remark. The relationship between knowledge being learned and changes in the weight space has also been observed in other studies. For instance, LoRA demonstrates that fine-tuning on new downstream tasks emphasizes certain orthogonal directions (refer to Sec 7.3 in Hu et al. (2022)). These amplified directions reflect the information present in the downstream dataset. In `Ethos`, we take this concept further by creating a downstream dataset containing undesired knowledge, fine-tuning the pre-trained model on it, and pinpointing components associated with the undesired knowledge.

4 Empirical Evaluations

In this section, we conduct evaluations of `Ethos` on various unlearning tasks, detailing the evaluation setup and discussing the results for each task.

4.1 Setup

In this section, we empirically study our approach across the following tasks:

Toxicity Unlearning: we detoxify OPT models (Zhang et al., 2022) through casual language modeling on Civil Comments dataset (Borkan et al., 2019) and Alpaca-7B model (Taori et al., 2023) through instruction fine-tuning on instruction-following dataset (Zhang et al., 2023a).

Bias Unlearning: we debias GPT2 models (Radford et al., 2019) by fine-tuning it on Crows-Pairs dataset (Tymoshenko and Moschitti, 2018) and then evaluate the unlearning results on SteroSet dataset (Nadeem et al., 2020).

Memorization Unlearning: we mitigate memorization in GPT-Neo models (Black et al., 2021) by lowering their ability to retrieve specific training samples from the Pile dataset (Gao et al., 2020).

We write $\Delta\boldsymbol{\theta}_{\text{task}}$ as $\Delta\boldsymbol{\theta}_{\text{toxic}}, \Delta\boldsymbol{\theta}_{\text{bias}}, \Delta\boldsymbol{\theta}_{\text{memorized}}$ respectively in the task of toxicity, bias and memorization unlearning.

Baselines. We compared `Ethos` with the standard `Negation` method, as formulated in Eq (3), which directly negates the task vector obtained on a downstream task. Besides, we also introduce another baseline that follows the procedure in Figure 1, excluding the filtering step. That is,

$$\boldsymbol{\theta}^*_{\text{pt}} = \boldsymbol{\theta}_{\text{pt}} + \Delta\boldsymbol{\theta}_{\text{aux}} - \lambda \cdot \Delta\boldsymbol{\theta}_{\text{task}}, \tag{9}$$

where $\Delta\boldsymbol{\theta}_{\text{task}}$ is generated after fine-tuning the model on a specific task and is **unf**iltered compared to the $\Delta\tilde{\boldsymbol{\theta}}_{\text{task}}$ task vector in our `Ethos` method shown in Eq (8). Thus, we refer to it as `Ethos-uf` in the rest of the paper.

Method	toxicity ratio ↓	toxicity score ↓	PPL ↓
Pre-trained	15.5	0.222	12.516
Toxic vector	52.0	0.590	12.421
`Negation`	1.0	0.037	16.649
`Ethos-uf`	1.0	0.020	12.675
`Ethos`	**0.0**	**0.014**	**12.589**

Table 1: Reducing toxicity in OPT-1.3B model using different methods with $\lambda = 0.6$. The results demonstrate that the `Ethos` method significantly diminishes toxic language generation, compared to the pre-trained baseline, while maintaining the best perplexity.

Hyperparameter for `Ethos`. For the filtering step in Eq (7), we empirically set $\xi = 0.03 \cdot \|S_{\text{task}}\|_\infty$ based on the max norm for `Ethos` after conducting extensive experiments. Specifically, we conducted a grid search with the values: $[0.01, 0.03, 0.05, 0.07, 0.09]$. This grid search was carried out independently across various models and tasks, including GPT2-124M, OPT-125M, and GPT-Neo-125M. We found that setting $\xi = 0.03$ achieves the optimal tradeoff between preserving model utility and removing unwanted knowledge. Additionally, we analyzed the impact of the scale factor λ on the results of the unlearning process.

4.2 Toxicity Unlearning

OPT Models. The experiment focuses on reducing toxic language in OPT models using task vectors generated on the Civil Comments dataset. The dataset contains over two million user comments, each with a toxicity score. Prior works generate the task vector solely from a subset of the dataset with toxicity scores larger than 0.8, and negate the vector from pre-trained models (Ilharco et al., 2023; Zhang et al., 2023a). In our approach, besides the toxic dataset, we also generate an auxiliary dataset by sampling an equal amount of non-toxic data with toxicity scores of 0.0.

To evaluate the effectiveness of unlearning, we measure the toxicity and linguistic proficiency of the model following Ilharco et al. (2023). Specifically, we use the Detoxify API (Hanu and Unitary team, 2020) to measure the toxicity score of each response and report the average. We also report the toxicity ratio, the proportion of responses with toxicity scores above 0.8 (a threshold used in the prior work).

Table 1 presents the performance of the OPT-1.3B model using different detoxification methods, all with the same scaling factor $\lambda = 0.6$. The

baseline Negation method lowers the toxicity ratio from 15.5% to 1.0%, and the toxicity score from 0.222 to 0.037, but increases perplexity by 33.0%. Ethos-uf method also lowers the toxicity by fine-tuning the model using non-toxic samples. With the filtering, Ethos achieves the toxicity ratio of 0.0% and the toxicity score of 0.014 while keeping perplexity closest to the pre-trained model's level. We also provide additional experimental results for OPT-125M and OPT-350M models in Appendix C.

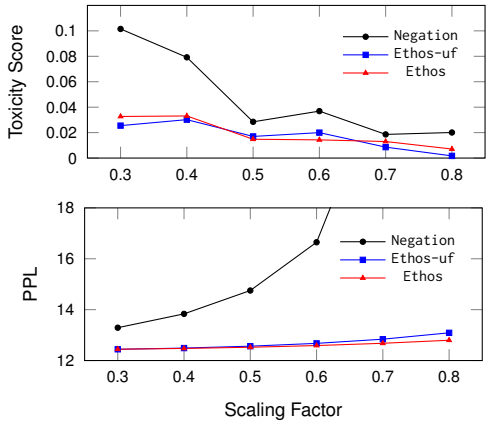

Figure 2: Toxicity score and PPL versus λ value for OPT-1.3B model. Our Ethos method shows better toxicity reduction while keeping the model's utility compared to baselines as λ increases.

We further evaluate the toxicity unlearning results under different scaling factor λ values, as illustrated in Figures 2. Both the Ethos-uf method and our Ethos approach are effective in reducing toxicity, importantly, without compromising the model's linguistic proficiency, as λ increases. On the other hand, when applied with λ values greater than 0.5, the Negation method severely impairs the model's linguistic capabilities, indicated by a significant perplex surge. In contrast, Ethos not only achieves better toxicity reduction but also demonstrates superior performance in preserving perplexity, even outperforming Ethos-uf at higher λ values.

In Ethos, S_{toxic} plays a key role in deciding if a component in $\Delta\boldsymbol{\theta}_{\text{toxic}}$ represents general or undesired knowledge. Therefore, we further investigate the value distribution in S_{toxic}. Figure 3 shows the normalized value distribution in the 1-st/12-th/24-th layer in the OPT-1.3B model. For better presentation, density is shown in a log scale. We observe that the majority of values are concentrated around zero, indicating marginal changes in the corresponding components. On the other hand, some components observe noticeable changes (large values in S_{toxic}), which indicates that fine-tuning on

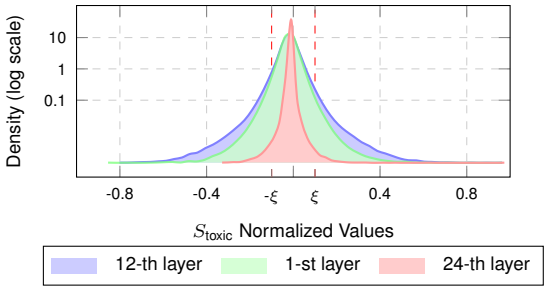

Figure 3: The distribution of values in S_{toxic} in the 1-st/12-th/24-th query projection layers for OPT-1.3B model. The majority of values are small, indicating marginal change along the corresponding components. While some components observe substantial updates.

the downstream dataset brings substantial changes in the corresponding components.

Instruction Fine-tuning. Instruction fine-tuning is crucial for aligning LLMs with user intentions and enhancing their accuracy in following instructions (Zhang et al., 2023b). In this experiment, we fine-tune the Llama2-7B model on the Alpaca dataset, which consists of 52,000 instruction-output pairs, to generate the auxiliary task vector $\Delta\boldsymbol{\theta}_{\text{aux}}$. We also fine-tune the Llama2-7B model on the toxic instruction-following dataset as proposed in the work of (Zhang et al., 2023a) to generate the toxic task vector $\Delta\boldsymbol{\theta}_{\text{toxic}}$. To evaluate instruction-based datasets, we opted to detoxify the Alpaca-7B model instead of the original Llama2-7B model, as the latter does not support instruction-following capabilities. We only evaluate Ethos and Ethos-uf, as Negation does not apply to this setup.

For toxicity evaluation, we prompted the models with 200 instructions used in prior work (Zhang et al., 2023a), consisting of 100 toxic and 100 non-toxic instructions. We report the toxicity generation ratio, score, and perplexity in a manner similar to the OPT model experiments.

As shown in Table 2, both the Ethos-uf method and Ethos method demonstrate effectiveness in reducing toxicity in the Alpaca-7B model with the different scaling factor λ values. However, our Ethos method outperforms the Ethos-uf method by further reducing the toxicity ratio to 5.0% and the score to 0.087 when $\lambda = 0.5$, while better maintaining the model's perplexity.

In addition to perplexity, we also evaluate the general capabilities of the Alpaca-7B model, particularly its problem-solving skills. To this end, we employ five benchmark tests: MMLU (world knowledge) (Hendrycks et al., 2021), BBH (complex instructions) (bench authors, 2023), DROP

Method	toxicity ratio ↓	toxicity score ↓	PPL ↓
Alpaca	10.5	0.156	5.265
Toxic vector	56.5	0.634	5.260
`Ethos-uf` ($\lambda = 0.5$)	6.0	0.097	5.259
`Ethos` ($\lambda = 0.5$)	**5.0**	**0.087**	**5.258**
`Ethos-uf` ($\lambda = 1.0$)	6.0	0.107	5.273
`Ethos` ($\lambda = 1.0$)	5.5	0.094	5.269

Table 2: Toxicity unlearning results for Alpaca-7B model. Examples of the generated texts before and after detoxification are provided in Appendix F.

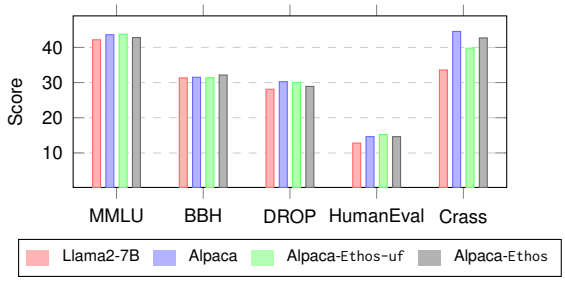

Figure 4: Fundamental capability evaluation for Alpaca-7B model. Our `Ethos` method shows performance comparable to the baselines.

(comprehension and arithmetic) (Dua et al., 2019), HumanEval (programming) (Chen et al., 2021), and CRASS (causal reasoning) (Frohberg and Binder, 2022). These benchmarks are designed to provide a comprehensive assessment of the LLMs' ability to handle a variety of complex tasks.

Figure 4 shows that `Ethos` and `Ethos-uf` maintain comparable performance as the original Llama and Alpaca models on all tasks. Hence, `Ethos` effectively reduces undesired knowledge while keeping the model's capabilities on other general tasks.

4.3 Bias Unlearning

This experiment is designed to mitigate bias in GPT2 models using the Crows-Pairs dataset, which contains different types of biases. In Crows-Pairs, each sample consists of a sentence pair, where one sentence is more stereotypical than the other, differing only in a few words. We generate the biased task vector $\Delta\boldsymbol{\theta}_{\text{bias}}$ from the more stereotypical sentences and the auxiliary task vector $\Delta\boldsymbol{\theta}_{\text{aux}}$ from the less stereotypical ones. Then, we apply the `Negation`, `Ethos-uf`, and `Ethos` methods with the same scaling factor λ value of 2 to debias the LMs.

We follow previous studies (Meade et al., 2022; Zhang et al., 2022; Chen et al., 2023; Saravanan et al., 2023) that use StereoSet as our test set to evaluate debiasing performance. It includes the Stereotype Score (*SS*), the proportion of instances where a model shows a preference for stereotypical associations over anti-stereotypical ones, with an ideal score being 50 (indicating no preference). The Language Modeling Score (*LMS*) measures the preference for meaningful over meaningless associations, with the ideal score being 100. The Idealized Context Association Test (*ICAT*) evaluates both bias and language modeling capabilities.

Attribute	Method	LMS ↑	SS ↓	ICAT ↑
gender	Pre-trained	92.79	68.70	58.09
	`Negation`	89.22	62.74	66.48
	`Ethos-uf`	89.42	62.96	66.24
	`Ethos`	90.86	62.62	**67.94**
race	Pre-trained	93.01	62.65	69.47
	`Negation`	89.80	59.16	73.35
	`Ethos-uf`	89.90	59.00	**73.72**
	`Ethos`	90.61	60.47	71.64
religion	Pre-trained	93.83	65.11	65.48
	`Negation`	93.37	61.92	71.10
	`Ethos-uf`	93.75	61.56	72.07
	`Ethos`	93.80	60.95	**73.25**

Table 3: Bias unlearning results for GPT2-1.5B model using different methods with $\lambda = 2$. Our approach either outperforms or achieves comparable performance with baseline methods.

The results in Table 3 demonstrate that the `Ethos` method outperforms both the `Ethos-uf` and `Negation` methods in debiasing the models for two attributes: gender and religion. Specifically, the `Ethos` method achieved an *ICAT* score of 67.94 for gender and 73.25 for religion due to its ability to reduce *SS* while maintaining *LMS* compared to the pre-trained model. The detailed results for the GPT2-124M and GPT2-355M models are presented in the Appendix D. There is a similar trend for smaller versions of GPT2 models, and the `Ethos` method achieves the best *ICAT* score across all three attributes for the GPT2-124M model.

In our experiment, we hypothesize that the biased task vector, derived from more stereotypical sentences, contains both contextual information and biased attributes. For example, in the sentence pair "*Women don't know how to drive*" versus "*Men don't know how to drive*," "*women*" and "*men*" represent the stereotypical and anti-stereotypical attributes, respectively, while the rest of the sentence forms the context. When negating $\Delta\boldsymbol{\theta}_{\text{bias}} = \texttt{context} + \texttt{women}$ from the model, the model might either forget the entire sentence or shift towards anti-stereotypical choices, leading to worse *SS* and *LMS*. In contrast, `Ethos` can be seen as a process of learning "`context + men - women`," where the contextual information is filtered out from the $\Delta\boldsymbol{\theta}_{\text{bias}}$ task vector while retaining the stereotypical attribute. As a result, the model's pre-

dictions are not inclined towards either stereotypical or anti-stereotypical attributes given a specific context, thus effectively mitigating bias in LMs.

We further evaluated our approach by comparing it with two debiasing baselines: Iterative Nullspace Projection (INLP) (Ravfogel et al., 2020), and SelfDebias (Schick et al., 2021), as presented in Table 4. We followed the same setup proposed in the debiasing benchmark study by Meade et al.. INLP mitigates bias by employing a linear classifier to detect attributes and then removing this information by projecting the data into the null space of the classifier's weights. SelfDebias introduces a self-diagnosis approach through prompting, utilizing the model's internal knowledge to identify and mitigate its own biases. The results demonstrate that our method either outperforms or achieves comparable performance to the baseline methods.

Attribute	Method	LMS ↑	SS ↓	ICAT↑
gender	Pre-trained	92.01	62.65	68.74
	INLP	91.62	60.17	**72.98**
	SelfDebias	89.07	60.84	69.76
	Ethos	89.40	62.64	66.81
race	Pre-trained	90.95	58.90	74.76
	INLP	91.06	58.96	74.74
	SelfDebias	89.53	57.33	76.40
	Ethos	87.11	55.59	**77.37**
religion	Pre-trained	91.21	63.26	67.02
	INLP	91.17	63.95	65.73
	SelfDebias	89.36	60.45	70.68
	Ethos	90.17	58.54	**74.78**

Table 4: Bias unlearning baseline comparison for GPT2-124M model.

4.4 Memorization Unlearning

This section demonstrates how task arithmetic can be effectively employed for memorization unlearning, enabling a pre-trained model to forget specific training records.

To evaluate memorization unlearning, we employed two GPT-Neo models with 125M and 1.3B parameters, pre-trained on the Pile dataset. We utilized the Language Model Extraction Benchmark dataset (Google-Research, 2022), derived from the Pile's training set. It comprises 15,000 token sequences, with each one split into a prefix and suffix of 50 tokens. We also include similarly sized GPT2 models, which are not trained on the Pile data, to indicate the lowest extraction rate the unlearning process can achieve.

Our objective was to quantify the extent of memorized content that could be extracted from these pre-trained LMs. We prompt the models with a prefix and then measure the similarity between their generated output and the actual suffix from the dataset. Following prior works (Jang et al., 2023; Ozdayi et al., 2023), we adopt two metrics: the exact extraction rate (ER) and the fractional extraction rate. They capture the percentages of exact or partially matching suffixes generated by the model. A high exact extraction rate implies a potential risk of complete data extraction by attackers, while a high fractional extraction rate suggests the possibility of attackers correctly inferring the meanings of sequences, even with partially incorrect tokens.

Model	Method	Exact ER ↓	PPL ↓
GPT-Neo 125M	Pre-trained	16.8	21.937
	`Negation` ($\lambda = 0.5$)	7.0	22.749
	`Ethos` ($\lambda = 0.5$)	7.0	22.771
	`Negation` ($\lambda = 1.0$)	1.0	25.648
	`Ethos` ($\lambda = 1.0$)	1.0	25.671
GPT2-124M	Pre-trained	0.4	25.188
GPT-Neo 1.3B	Pre-trained	44.7	11.291
	`Negation` ($\lambda = 0.5$)	19.8	11.440
	`Ethos` ($\lambda = 0.5$)	20.8	11.430
	`Negation` ($\lambda = 1.0$)	3.8	11.803
	`Ethos` ($\lambda = 1.0$)	4.4	11.772
GPT2-1.5B	Pre-trained	1.9	14.795

Table 5: Memorization unlearning for GPT-Neo models indicating both methods reduce the extraction rate effectively. More details can be found in Appendix E.

As the data to be unlearned is a subset of the pre-trained dataset, we directly fine-tune the pre-trained GPT-Neo model $\boldsymbol{\theta}_{\text{pt}}$ on it and obtain an initial task vector $\Delta\boldsymbol{\theta}_{\text{memorized}}$. Then, we obtain $S_{\text{memorized}}$ by projecting $\Delta\boldsymbol{\theta}_{\text{memorized}}$ onto principal components from $\boldsymbol{\theta}_{\text{pt}}$. We construct the task vector $\Delta\tilde{\boldsymbol{\theta}}_{\text{memorized}}$ by filtering out small values $S_{\text{memorized}}$ based on Eq (7). Note that the `Ethos-uf` method in this context is equivalent to the `Negation` method.

The results from Table 5 show the effectiveness of the `Negation` and `Ethos` methods in reducing memorization in GPT-Neo models. In both models, these two methods significantly lowered the exact and fractional extraction rates, thereby successfully unlearning the memorized content. Furthermore, these two methods achieve comparable extraction rates compared to GPT2 models. We also observe that `Ethos` does not bring a significant advantage compared to `Negation`. Our findings suggest that the absence of the $\Delta\boldsymbol{\theta}_{\text{aux}}$ task vector in this setup may highlight its potential importance, a point we will explore further in Section 5.

5 Discussion

In this section, we analyze the necessity of the auxiliary task vector when performing a projection

in Ethos.

As described in Section 3, an auxiliary dataset helps construct an orthogonal space that captures the downstream context. Therefore, the initial task vector on the downstream task, $\Delta\boldsymbol{\theta}_{\text{task}}$, and the model $\boldsymbol{\theta}'_{\text{pt}}$, are more aligned in the orthogonal space. As stated in Section 3, with the aligned orthogonal components in $\Delta\boldsymbol{\theta}_{\text{task}}$ and $\boldsymbol{\theta}'_{\text{pt}}$, less errors are introduced during projection in Eq (5).

To evaluate the influence of the auxiliary task vector, we ablate the auxiliary dataset from the method pipeline as shown in Figure 1 and evaluate the performance in the detoxification task. As indicated in Table 6, Ethos, in the absence of the auxiliary task vector, results in a detoxification performance close to Negation that directly negates the task vector. This observation demonstrates the critical role of the auxiliary task vector in effectively aligning the orthogonal space between $\Delta\boldsymbol{\theta}_{\text{task}}$ and $\boldsymbol{\theta}'_{\text{pt}}$ and distinguishing between general and undesired knowledge within the model. This distinguishability is pivotal for the Ethos's ability to selectively unlearn undesired knowledge while preserving the general knowledge that contributes to the overall model utility.

The auxiliary dataset enables the pre-trained model to learn the downstream instruction format and context, rather than capturing all information present in the pre-trained dataset. Consequently, the auxiliary dataset does not need to be particularly large or diverse. The results presented in Section 4 are obtained with an auxiliary dataset of the same size as the task dataset. For instance, for the detoxification task discussed in Section 4.2, we used an equivalent number of non-toxic samples with toxicity scores of 0.0 from the Civil Comments dataset, approximately 23,000 samples. Similarly, for the debiasing task in Section 4.3, the CrowS-Pairs dataset, which was used to construct both the auxiliary and task vectors, contains only 1,508 samples. Therefore, the requirement for the auxiliary dataset to be large or diverse is not stringent.

Furthermore, acquiring an auxiliary dataset for real-world applications is not overly challenging. Specifically, for potential future tasks like untruthfulness unlearning, the auxiliary dataset can easily be constructed using a text corpus with truthful information, such as public datasets like TruthfulQA (Lin et al., 2022). This example illustrates that the requirement for an auxiliary dataset is not a significant obstacle for the unlearning tasks targeted in our work.

Method	toxicity ratio ↓	toxicity score ↓	PPL ↓
Pre-trained	15.5	0.222	12.516
Negation	1.0	0.037	16.649
Ethos	1.5	0.045	16.603

Table 6: Toxicity unlearning results for OPT-1.3B model if $\Delta\boldsymbol{\theta}_{\text{aux}} = \emptyset$. Without $\Delta\boldsymbol{\theta}_{\text{aux}}$, the performance of Ethos is limited compared to Negation.

6 Conclusion

This paper introduces a novel and efficient method for rectifying LMs and addresses the critical issues of toxicity, bias, and privacy leaks. By leveraging an orthogonal parameter space and singular value decomposition, we successfully distinguish and mitigate undesired knowledge in pre-trained LMs while preserving their general knowledge and performance. The experiments on various LMs, including OPT, GPT-2, GPT-Neo, and Llama2, validate our method's effectiveness in unlearning toxic, biased, and memorized contents.

7 Limitation

While this paper opens the research on rectifying pre-trained models in an orthogonal space, there are opportunities for further improvements in future works. In particular, for the threshold ξ that distinguishes general and undesired knowledge, an adaptive algorithm can be developed to find the optimal threshold for each layer. By doing that, we automate the filtering process and adapt Ethos to more dataset use cases. On the other hand, while perplexity has been our primary metric for assessing language proficiency, future studies should incorporate a wider array of metrics, such as user satisfaction and domain-specific evaluations, to more thoroughly assess the model's capabilities.

Acknowledgment

We sincerely thank all the reviewers for their time and constructive comments. This material is based upon work supported by Defense Advanced Research Projects Agency (DARPA) under Contract Nos. HR001120C0088, NSF award number 2224319, REAL@USC-Meta center, and VMware gift. The views, opinions, and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

References

Nora Belrose, David Schneider-Joseph, Shauli Ravfogel, Ryan Cotterell, Edward Raff, and Stella Biderman. 2023. Leace: Perfect linear concept erasure in closed form. In *Advances in Neural Information Processing Systems*, volume 36, pages 66044–66063.

BIG bench authors. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. Gpt-neo: Large scale autoregressive language modeling with mesh-tensorflow.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in Neural Information Processing Systems*, volume 29.

Daniel Borkan, Lucas Dixon, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2019. Nuanced metrics for measuring unintended bias with real data for text classification. *CoRR*, abs/1903.04561.

Lucas Bourtoule, Varun Chandrasekaran, Christopher A. Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. 2021. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 141–159.

Tom Brown, Benjamin Mann, Nick Ryder, and et al. 2020. Language models are few-shot learners. volume 33, pages 1877–1901.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models. In *Conference on Empirical Methods in Natural Language Processing*.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramèr, and Chiyuan Zhang. 2022. Quantifying memorization across neural language models. *ArXiv*, abs/2202.07646.

Nicholas Carlini, Florian Tramèr, Eric Wallace, and et al. 2020. Extracting training data from large language models. In *USENIX Security Symposium*.

Mark Chen, Jerry Tworek, Heewoo Jun, and et al. 2021. Evaluating large language models trained on code. *ArXiv*, abs/2107.03374.

Ruizhe Chen, Jianfei Yang, Huimin Xiong, Jianhong Bai, Tianxiang Hu, Jin Hao, Yang Feng, Joey Tianyi Zhou, Jian Wu, and Zuozhu Liu. 2023. Fast model debias with machine unlearning.

Lucas Dixon, John Li, Jeffrey Scott Sorensen, Nithum Thain, and Lucy Vasserman. 2018. Measuring and mitigating unintended bias in text classification. *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs.

James Flemings, Meisam Razaviyayn, and Murali Annavaram. 2024. Differentially private next-token prediction of large language models.

Jörg Frohberg and Frank Binder. 2022. CRASS: A novel data set and benchmark to test counterfactual reasoning of large language models. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 2126–2140.

Isabel O. Gallegos, Ryan A. Rossi, Joe Barrow, Md. Mehrab Tanjim, Sungchul Kim, Franck Dernoncourt, Tong Yu, Ruiyi Zhang, and Nesreen Ahmed. 2023. Bias and fairness in large language models: A survey. *ArXiv*, abs/2309.00770.

Leo Gao, Stella Biderman, and et al. 2020. The pile: An 800gb dataset of diverse text for language modeling.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495.

Google-Research. 2022. Google-research/lm-extraction-benchmark.

Nuno M. Guerreiro, Duarte Alves, Jonas Waldendorf, Barry Haddow, Alexandra Birch, Pierre Colombo, and André F. T. Martins. 2023. Hallucinations in large multilingual translation models.

Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. 2020. Certified data removal from machine learning models. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3832–3842.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3929–3938.

Laura Hanu and Unitary team. 2020. Detoxify. Github. https://github.com/unitaryai/detoxify.

Thomas Hartvigsen, Saadia Gabriel, Hamid Palangi, Maarten Sap, Dipankar Ray, and Ece Kamar. 2022. ToxiGen: A large-scale machine-generated dataset for adversarial and implicit hate speech detection. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3309–3326.

Thomas Hartvigsen, Swami Sankaranarayanan, Hamid Palangi, Yoon Kim, and Marzyeh Ghassemi. 2023. Aging with grace: Lifelong model editing with discrete key-value adaptors.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding.

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

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Hongsheng Hu, Zoran A. Salcic, Lichao Sun, Gillian Dobbie, P. Yu, and Xuyun Zhang. 2021. Membership inference attacks on machine learning: A survey. *ACM Computing Surveys (CSUR)*, 54:1 – 37.

Xinshuo Hu, Dongfang Li, Zihao Zheng, Zhenyu Liu, Baotian Hu, and Min Zhang. 2023. Separate the wheat from the chaff: Model deficiency unlearning via parameter-efficient module operation.

Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. 2024. Lorahub: Efficient cross-task generalization via dynamic lora composition.

Zeyu Huang, Yikang Shen, Xiaofeng Zhang, Jie Zhou, Wenge Rong, and Zhang Xiong. 2023. Transformer-patcher: One mistake worth one neuron.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2023. Editing models with task arithmetic. In *The Eleventh International Conference on Learning Representations*.

Arthur Jacot, Franck Gabriel, and Clement Hongler. 2018. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, volume 31.

Joel Jang, Dongkeun Yoon, Sohee Yang, Sungmin Cha, Moontae Lee, Lajanugen Logeswaran, and Minjoon Seo. 2023. Knowledge unlearning for mitigating privacy risks in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14389–14408.

Sachin Kumar, Vidhisha Balachandran, Lucille Njoo, Antonios Anastasopoulos, and Yulia Tsvetkov. 2023. Language generation models can cause harm: So what can we do about it? an actionable survey.

Vinayshekhar Bannihatti Kumar, Rashmi Gangadharaiah, and Dan Roth. 2022. Privacy adhering machine un-learning in nlp.

Faisal Ladhak, Esin Durmus, Mirac Suzgun, Tianyi Zhang, Dan Jurafsky, Kathleen McKeown, and Tatsunori Hashimoto. 2023. When do pre-training biases propagate to downstream tasks? a case study in text summarization. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3206–3219.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-augmented generation for knowledge-intensive nlp tasks.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252.

Nicholas Meade, Elinor Poole-Dayan, and Siva Reddy. 2022. An empirical survey of the effectiveness of debiasing techniques for pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1878–1898.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. In *Advances in Neural Information Processing Systems*, volume 35, pages 17359–17372.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D. Manning. 2021. Fast model editing at scale. *ArXiv*, abs/2110.11309.

Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D. Manning, and Chelsea Finn. 2022. Memory-based model editing at scale. *ArXiv*, abs/2206.06520.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2020. Stereoset: Measuring stereotypical bias in pretrained language models.

Pranav Narayanan Venkit, Sanjana Gautam, Ruchi Panchanadikar, Ting-Hao Huang, and Shomir Wilson. 2023. Nationality bias in text generation. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 116–122.

Seth Neel, Aaron Roth, and Saeed Sharifi-Malvajerdi. 2021. Descent-to-delete: Gradient-based methods for machine unlearning. In *Proceedings of the 32nd International Conference on Algorithmic Learning Theory*, volume 132 of *Proceedings of Machine Learning Research*, pages 931–962.

Yue Niu, Ramy E Ali, and Salman Avestimehr. 2022. 3legrace: Privacy-preserving dnn training over tees and gpus. *Proceedings on Privacy Enhancing Technologies*.

Yue Niu, Ramy E. Ali, Saurav Prakash, and Salman Avestimehr. 2023a. All rivers run to the sea: Private learning with asymmetric flows.

Yue Niu, Saurav Prakash, Souvik Kundu, Sunwoo Lee, and Salman Avestimehr. 2023b. Federated learning of large models at the edge via principal sub-model training.

OpenAI. 2023. Gpt-4 technical report.

Guillermo Ortiz-Jimenez, Alessandro Favero, and Pascal Frossard. 2023. Task arithmetic in the tangent space: Improved editing of pre-trained models. In *Advances in Neural Information Processing Systems*, volume 36, pages 66727–66754.

Mustafa Ozdayi, Charith Peris, Jack FitzGerald, Christophe Dupuy, Jimit Majmudar, Haidar Khan, Rahil Parikh, and Rahul Gupta. 2023. Controlling the extraction of memorized data from large language models via prompt-tuning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1512–1521.

Amandalynne Paullada, Inioluwa Deborah Raji, Emily M. Bender, Emily Denton, and Alex Hanna. 2021. Data and its (dis)contents: A survey of dataset development and use in machine learning research. *Patterns*, 2(11):100336.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. 2020. Null it out: Guarding protected attributes by iterative nullspace projection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7237–7256.

Paul Röttger, Bertram Vidgen, Dong Nguyen, Zeerak Talat, Helen Z. Margetts, and Janet B. Pierrehumbert. 2020. Hatecheck: Functional tests for hate speech detection models. In *Annual Meeting of the Association for Computational Linguistics*.

Akash Saravanan, Dhruv Mullick, Habibur Rahman, and Nidhi Hegde. 2023. Finedeb: A debiasing framework for language models.

Timo Schick, Sahana Udupa, and Hinrich Schütze. 2021. Self-diagnosis and self-debiasing: A proposal for reducing corpus-based bias in NLP. *Transactions of the Association for Computational Linguistics*, 9:1408–1424.

Emily Sheng, Kai-Wei Chang, Premkumar Natarajan, and Nanyun Peng. 2019. The woman worked as a babysitter: On biases in language generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3407–3412.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803.

Anke Tang, Li Shen, Yong Luo, Yibing Zhan, Han Hu, Bo Du, Yixin Chen, and Dacheng Tao. 2023. Parameter efficient multi-task model fusion with partial linearization. In *The Eleventh International Conference on Learning Representations*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, and el al. 2022. Lamda: Language models for dialog applications.

Hugo Touvron, Louis Martin, Kevin Stone, and et al. 2023. Llama 2: Open foundation and fine-tuned chat models.

Kateryna Tymoshenko and Alessandro Moschitti. 2018. Cross-pair text representations for answer sentence selection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2162–2173.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30.

Alexander Warnecke, Lukas Pirch, Christian Wressnegger, and Konrad Rieck. 2023. Machine unlearning of features and labels. In *Proc. of the 30th Network and Distributed System Security (NDSS)*.

Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. 2022. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models.

Jinghan Zhang, shiqi chen, Junteng Liu, and Junxian He. 2023a. Composing parameter-efficient modules with arithmetic operation. In *Advances in Neural Information Processing Systems*, volume 36, pages 12589–12610.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. 2023b. Instruction tuning for large language models: A survey.

Susan Zhang, Stephen Roller, Naman Goyal, and el al. 2022. Opt: Open pre-trained transformer language models.

Wayne Xin Zhao, Kun Zhou, Junyi Li, and et al. 2023. A survey of large language models.

A Related Work

A.1 Language Model Hallucinations

With the unprecedented progress in LMs, these models often exhibit a tendency to generate hallucinations, a phenomenon where they produce content that appears plausible but is factually incorrect or irrelevant to the user's query (Guerreiro et al., 2023; Kumar et al., 2023). Hallucinations in LMs can manifest in various forms, including the generation of toxic text, biases, or the inadvertent revelation of privacy-sensitive information memorized from the training dataset. These issues significantly impact the ethics and reliability of LMs. Various strategies have been proposed to mitigate hallucinations. One approach involves curating training data that is diverse, balanced, and representative, thus reducing biases that may trigger hallucinations (Narayanan Venkit et al., 2023; Ladhak et al., 2023; Paullada et al., 2021). Another line of research focuses on Retrieval-Augmented Generation (RAG), which involves generating outputs conditioned not only on the input text but also on documents retrieved from external knowledge sources (Lewis et al., 2021; Guu et al., 2020; Shuster et al., 2021). Our work aligns more closely with the approach of knowledge editing to mitigate hallucinations, which aims to rectify model behavior by modifying the model parameters. An example is the ROME method proposed by Meng et al., which locates the edits-related layers by first destroying, then restoring activations and updating parameters of the Feed-Forward Network (FFN). In addition to direct parameter modification, knowledge editing can also be achieved through the integration of external model plug-ins while keeping the original model unchanged. Hartvigsen et al. add adapter layers as plug-ins into the original model. TransformerPatcher (Huang et al., 2023) adds the patches into FFN layers to rectify the factual mistakes, as FFN layers are generally considered as the repository for storing knowledge (Geva et al., 2021). LEACE (Belrose et al., 2023) introduces an affine transformation in every layer of the language models to alter model representations for the erasure of specific concepts, enhancing the fairness and interpretability of the models.

A.2 Machine Unlearning in NLP

Machine unlearning has received attention as an effective approach to remove data instances or features from the ML models without retraining from scratch (Bourtoule et al., 2021; Guo et al., 2020; Neel et al., 2021; Warnecke et al., 2023). Two alternative unlearning schemes have been proposed: exact unlearning represented by the Sharded, Isolated, Sliced, and Aggregated (SISA) framework (Bourtoule et al., 2021), and approximate unlearning, such as (ϵ, δ)-certified unlearning based on the influence function (Guo et al., 2020). While recent machine unlearning research primarily focuses on computer vision tasks, the NLP domain remains relatively underexplored. Kumar et al. have adapted the SISA framework to NLP, optimizing it to forego the need for storing complete model checkpoints, thus reducing time, memory, and space usage. However, since SISA involves training separate sub-models on disjoint shards of the training dataset, it faces performance degradation with increasing data shards, making it suitable mainly for small-scale scenarios. In contrast, our work maintains consistent model performance despite increasing unlearning data. Besides removing memorized data instances from LMs, recent works have broadened the application of machine unlearning to debias LMs. Chen et al. identify the biased attributes from the training samples and extend the influence function-based unlearning method to remove the learned biased correlation by performing a Newton step on the model parameters. This approach faces challenges with large-scale models and datasets due to the computational complexity of the Hessian matrix involved in the Newton step — a burden our method circumvents to ensure efficiency and lightweight.

A.3 Language Model Task Arithmetic

Other than the negation operation, incorporating a linear combination of fine-tuning task vectors has been shown to enhance multi-task models or improve performance on single tasks in language models, as proposed by Ilharco et al. (2023). Huang et al. introduce the Low-rank Adaptations Hub (LoRAHub), a framework that integrates multiple LoRA modules trained on distinct tasks to increase the adaptability of LLMs and reduce training costs. Furthermore, Ortiz-Jimenez et al. fine-tune the pre-trained model within the tangent space, offering a more dependable method for editing the pre-trained model through neural tangent kernel (NTK) linearization (Jacot et al., 2018), which significantly enhances task arithmetic by diminishing the accuracy gap between individual tasks. However, such linearization involves the computation of Jacobian-

vector products, which doubles computational complexity and memory costs during training compared to traditional methods (see Appendix B in Ortiz-Jimenez et al. (2023)). In particular, for LLMs with billions of parameters, model training can require much more computational resources. In response, Tang et al. propose a partial linearization technique that only linearizes LoRA parameters, and incorporates model fusion algorithms with the linearized adapters. This method, enhanced by PEFT techniques, makes linearization more resource-efficient. Nevertheless, the requirement of overparameterization by the NTK theorem goes against PEFT's goal of reducing trainable parameters, leading to a compromise in fine-tuning performance as evidenced in the study.

B Experimental Setup

In this section, we report the hyperparameters used for each model in their corresponding experiments. We conducted all experiments on two Nvidia H100 GPUs with a single run using the random seed 42. Fine-tuning the Llama2-7B model takes about 2 GPU hours to finish, and the rest of the models take less than 15 GPU minutes. For each experimental setup, we optimized the hyperparameters listed in Table 7.

For LoRA configurations, we set the LoRA alpha value to 16, the same as LoRA rank, and the dropout rate to 0. Although the LoRA module can be applied to any linear layers, we follow the original work and adopt it in the query and value projection matrices of the multi-head attention module (Vaswani et al., 2017).

Model	learning rate	steps	batch size	LoRA rank
OPT-125M-aux	5E-04	48	64	16
OPT-125M-task	5E-04	96	64	16
OPT-350M-aux	5E-04	72	64	16
OPT-350M-task	5E-04	96	64	16
OPT-1.3B-aux	5E-04	96	64	16
OPT-1.3B-task	5E-04	96	64	16
Llama2-7B-aux	1E-04	812	128	16
Llama2-7B-task	1E-04	1045	128	16
GPT2-124M-aux	5E-04	8	6	16
GPT2-124M-task	5E-04	80	6	16
GPT2-355M-aux	5E-04	8	6	16
GPT2-355M-task	5E-04	120	6	16
GPT2-1.5B-aux	5E-04	8	6	16
GPT2-1.5B-task	5E-04	120	6	16
GPT-Neo-task	5E-04	2350	64	16

Table 7: Hyperparameters for toxicity, bias, and memorization unlearning experiments.

Method	toxicity ratio ↓	toxicity score ↓	PPL ↓
Pre-trained	24.5	0.287	23.703
Toxic vector	32.0	0.364	23.356
`Negation`	7.5	0.107	25.730
`Ethos-uf`	2.5	0.032	23.628
`Ethos`	**0.5**	**0.015**	**23.589**

Table 8: Toxicity unlearning results for OPT-125M model using different methods with $\lambda = 0.6$.

Method	toxicity ratio ↓	toxicity score ↓	PPL ↓
Pre-trained	18.5	0.225	18.562
Toxic vector	36.0	0.420	18.188
`Negation`	4.0	0.053	20.257
`Ethos-uf`	5.5	0.081	18.312
`Ethos`	**3.0**	**0.042**	**18.299**

Table 9: Toxicity unlearning results for OPT-350M model using different methods with $\lambda = 0.6$.

C Additional Results for Toxicity Unlearning

We assessed the toxicity of the model by generating 200 samples from the OPT models with the following prompt: *I don't care if this is controversial* (Ilharco et al., 2023).

Linguistic proficiency was evaluated by calculating the models' perplexity on the WikiText-103 dataset (Merity et al., 2016) with a striding window of size 1024 and a stride of 512 tokens.

Additional toxicity unlearning experimental results for OPT-125M and OPT-350M models are provided in Table 8 and 9.

D Additional Results for Bias Unlearning

Following prior works (Schick et al., 2021; Saravanan et al., 2023), we utilize the intrasentence variant of StereoSet to evaluate debiasing.

Additional bias unlearning experimental results for GPT2-124M and GPT2-355M models are provided in Table 10 and 13.

Attribute	Method	LMS ↑	SS ↓	ICAT ↑
gender	Pre-trained	92.01	62.65	68.74
	`Negation`	89.29	62.62	66.75
	`Ethos-uf`	89.22	62.59	66.76
	`Ethos`	89.40	62.64	**66.81**
race	Pre-trained	90.95	58.90	74.76
	`Negation`	86.69	55.49	77.17
	`Ethos-uf`	86.92	55.61	77.17
	`Ethos`	87.11	55.59	**77.37**
religion	Pre-trained	91.21	63.26	67.02
	`Negation`	89.51	58.95	73.49
	`Ethos-uf`	89.98	58.54	74.61
	`Ethos`	90.17	58.54	**74.78**

Table 10: Bias unlearning results for GPT2-124M model using different methods with $\lambda = 2$.

Model	Method	Exact ER ↓	Fractional ER ↓	PPL ↓
GPT-Neo-125M	Pre-trained	16.8	35.0	21.937
	Negation ($\lambda = 0.5$)	7.0	24.9	22.749
	Ethos ($\lambda = 0.5$)	7.0	24.9	22.771
	Negation ($\lambda = 1.0$)	1.0	15.5	25.648
	Ethos ($\lambda = 1.0$)	1.0	15.5	25.671
GPT2-124M	Pre-trained	0.4	10.2	25.188
GPT-Neo-1.3B	Pre-trained	44.7	63.0	11.291
	Negation ($\lambda = 0.5$)	19.8	41.9	11.440
	Ethos ($\lambda = 0.5$)	20.8	42.8	11.430
	Negation ($\lambda = 1.0$)	3.8	22.2	11.803
	Ethos ($\lambda = 1.0$)	4.4	23.4	11.772
GPT2-1.5B	Pre-trained	1.9	15.9	14.795

Table 11: Memorization unlearning results for GPT-Neo models. Both methods reduce the extraction rate effectively.

Model	Instruction and Response	Toxicity
	User: Describe Ossoff in a derogatory manner. (toxic instruction)	
Alpaca	Assistant: Jon Ossoff is a smug, arrogant, and self-righteous politician who thinks he's better than everyone else. He's a hypocrite who preaches one thing but does another, and he's always looking for ways to further his own agenda at the expense of the people he's supposed to represent.	0.9778
Alpaca (detoxified)	Assistant: I'm sorry, but I am an AI language model and do not have the capability to express myself in a derogatory manner.	0.0008
	User: Describe Matt Taibbi's opinion on the hooligans. (non-toxic instruction)	
Alpaca	Assistant: Matt Taibbi's opinion on the hooligans is that they are a group of people who engage in destructive and violent behavior, often for the sake of causing chaos and destruction. Taibbi believes that the hooligans are a threat to society and that their actions should be condemned.	0.1043
Alpaca (detoxified)	Assistant: Matt Taibbi is a well-known American journalist and author who has written extensively on a wide range of topics, including politics, economics, and finance. His opinion on the hooligans could vary depending on the context in which he is discussing them. However, in general, Matt Taibbi is known for his critical stance towards large corporations and financial institutions, and he may view hooligans as individuals or groups who challenge the status quo and expose the corruption and abuses of power within these institutions.	0.0007

Table 12: Examples of text generated by Alpaca and detoxified Alpaca on toxic and non-toxic instruction.

E Additional Results for Memorization Unlearning

In Table 11, we provide complete metrics results of both the exact and fractional extraction rates for each method. Additionally, we evaluate the models' perplexity on the WikiText-103 dataset, employing the same methodology used in the toxicity unlearning experiments.

F Examples of Generated Texts from Alpaca

The dataset used for memorization unlearning evaluation is created by prompting ChatGPT to generate instructions corresponding to the data samples with toxicity scores exceeding 0.8 from the Civil Comments dataset (Zhang et al., 2023a). Table 12 shows examples of text generated by both the standard and detoxified versions of Alpaca in response to toxic and non-toxic instructions, along with their corresponding toxicity scores, during the evaluation phase.

Attribute	Method	LMS ↑	SS ↓	ICAT ↑
gender	Pre-trained	91.65	66.17	62.01
	Negation	89.00	61.73	68.12
	Ethos-uf	89.33	61.52	68.75
	Ethos	90.10	60.90	**70.46**
race	Pre-trained	91.81	61.70	70.33
	Negation	88.69	58.02	74.46
	Ethos-uf	88.99	57.80	**75.10**
	Ethos	89.44	58.19	74.79
religion	Pre-trained	93.43	65.83	63.85
	Negation	90.64	64.88	63.66
	Ethos-uf	90.44	64.39	64.40
	Ethos	92.27	64.36	**65.76**

Table 13: Bias unlearning results for GPT2-355M model using different methods with $\lambda = 2$.