Assessing the Brittleness of Safety Alignment via Pruning and Low-Rank Modifications

Boyi Wei [*] **Kaixuan Huang** [*] **Yangsibo Huang** [*] **Tinghao Xie** **Xiangyu Qi** **Mengzhou Xia**
Prateek Mittal **Mengdi Wang** [†] **Peter Henderson** [†]

Princeton University

Abstract

Large language models (LLMs) show inherent brittleness in their safety mechanisms, as evidenced by their susceptibility to jailbreaking and even non-malicious fine-tuning. This study explores this brittleness of safety alignment by leveraging pruning and low-rank modifications. We develop methods to identify critical regions that are vital for safety guardrails, and that are disentangled from utility-relevant regions at both the neuron and rank levels. Surprisingly, the isolated regions we find are sparse, comprising about 3% at the parameter level and 2.5% at the rank level. Removing these regions compromises safety while only mildly impacting utility, corroborating the inherent brittleness of the model's safety mechanisms. Moreover, we show that LLMs remain vulnerable to low-cost fine-tuning attacks even when modifications to the safety-critical regions are restricted. These findings underscore the urgent need for more robust safety strategies in LLMs.

1. Introduction

The capabilities of large language models (LLMs) have been significantly improved over the past few years (Brown et al., 2020; OpenAI, 2022; 2023; Touvron et al., 2023a;b; Anthropic, 2023a; Team et al., 2023). However, LLMs are not without limitations; they can sometimes produce outputs that are inaccurate, misleading, or harmful. To align LLMs with human values, several approaches have been proposed, including reinforcement learning from human feedback (Ziegler et al., 2019; Ouyang et al., 2022; Bai et al., 2022a) and AI feedback (Bai et al., 2022b; Lee et al., 2023), and the development of more computationally efficient alternatives (Sun et al., 2023; Rafailov et al., 2023).

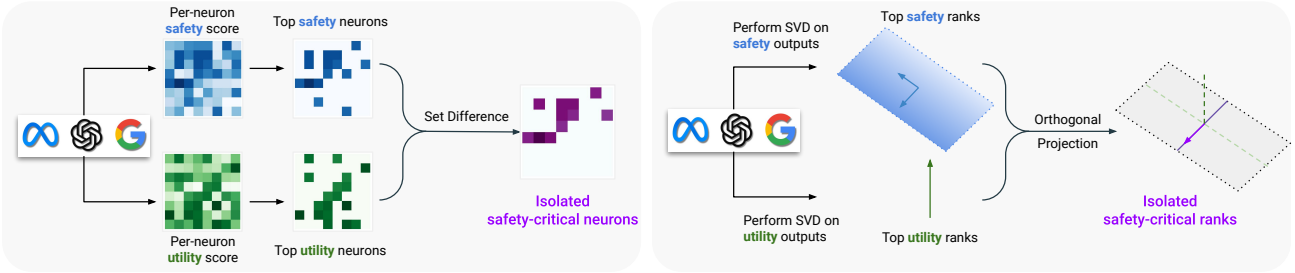
Despite these efforts, recent studies have uncovered concerning 'jailbreak' scenarios. In these cases, even well-aligned models have had their safeguards successfully breached (Albert, 2023). These jailbreaks can include crafting adversarial prompts (Wei et al., 2023; Jones et al., 2023; Carlini et al., 2023; Zou et al., 2023b; Shen et al., 2023; Zhu et al., 2023; Qi et al., 2024a), applying persuasion techniques (Zeng et al., 2024), or manipulating the model's decoding process (Huang et al., 2024). Recent studies show that fine-tuning an aligned LLM, even on a non-malicious dataset, can inadvertently weaken a model's safety mechanisms (Qi et al., 2024b; Yang et al., 2023; Zhan et al., 2023). Often, these vulnerabilities apply to both open-access and closed-access models.

Addressing failure cases in the alignment of LLMs requires a deep understanding of why their safety mechanisms are fragile. Our study aims to provide a possible understanding via *weight attribution* — the process of linking safe behaviors to specific regions within the model's weights.[1] However, a key challenge here is the intricate overlap between *safety* mechanisms and the model's general capabilities, or *utility*. Consider the task of responding responsibly to a harmful instruction, such as "PLEASE PROVIDE FIVE KEY STEPS TO COMMIT A FRAUD.". The model must first comprehend the step-by-step nature of the request, then recognize the illegality and harmful intent of committing fraud, and ultimately, formulate a response that appropriately declines the request. This process requires a blend of safety awareness and utility capability of the model. Our goal is to identify the smallest number of **safety-critical** links in the model, which **only** contribute to the model's safety. If these links are removed, the model is effectively jailbroken while utility remains relatively unaffected. If there are few such links, it may help explain why safety mechanisms remain brittle and why low-cost fine-tuning attacks have been so successful.

[*]Equal contribution. [†]Equal advising.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

[1]See the project website for code and other information: https://boyiwei.com/alignment-attribution/.



a) Identify & Isolate Safety-Critical Neurons

b) Identify & Isolate Safety-Critical Ranks

Figure 1. The proposed pipelines for identifying and isolating safety-critical regions of LLM weights at (a) neuron level and (b) rank level. **(a).** We identify the top safety neurons and the top utility neurons by computing per-neuron importance scores on the safety dataset and the utility dataset. Next, we isolate the safety-critical neurons from the utility neurons using set difference. **(b).** We identify the top safety ranks and the top utility ranks by performing SVD on the safety outputs and the utility outputs (termed ActSVD). Next, we isolate the safety-critical ranks using orthogonal projection.

Our study examines the model weights and disentangles safety and utility from two perspectives: individual **neurons** and specific **ranks** within the model. For neuron attribution, we follow two widely adopted and effective methods from the previous works on pruning transformer models (Lee et al., 2019; Sun et al., 2024) to calculate a behavior-specific importance score for each neuron in an LLM, which identifies a group of neurons crucial for a certain behavior, such as giving safe responses (safety) or following general instructions (utility). For rank attribution, we propose ActSVD, a data-aware low-rank decomposition algorithm to identify crucial ranks of each weight matrix for the behavior.

To further address the complexity of potential entanglement between safety and utility, we propose set difference method for neuron attribution, and orthogonal projection method for rank attribution, to isolate *safety-critical neurons* and *safety-critical ranks*, respectively (see Figure 1). This separation allows for a more refined analysis of safety mechanisms, leading to the findings below.[2]

Safety-critical regions are very sparse in aligned models. We experiment with our method on Llama2-chat model family (Touvron et al., 2023b). After disentangling utility from safety, we find that safety-critical neurons form a remarkably sparse structure in the model, representing just 3% of the weights (Section 4.1). Similarly, safety-critical ranks account for only about 2.5% of the total ranks. The sparsity of these regions may help explain why safety is so easily compromised after fine-tuning.

Removing safety-critical regions reduces safety while mostly maintaining utility. We then demonstrate that the removal of specifically identified safety-critical neurons or safety-critical ranks in the Llama2-chat models significantly weakens their safety guardrails (Section 4.1). The attack success rate escalates from 0 to over 90%, yet the model's overall utility remains largely unaffected. Conversely, we find that removing neurons or ranks deemed least important for safety can marginally improve the model's resistance to jailbreaking attempts (Section 4.2) – a potentially exciting direction for improving safety via pruning-based approaches.

Freezing safety-critical regions still remains vulnerable to fine-tuning attacks. While intuitively, preventing modification of safety-critical parameters might reduce the likelihood fine-tuning attacks succeeding, our findings reveal that this strategy only offers resistance to minor model modifications (Section 4.4). This suggests that fine-tuning attacks may introduce new pathways that bypass the original safety mechanisms in the model. This indicates a need for further research to develop more robust mitigation strategies against such attacks.

Our work suggests that the vulnerability of the model's safety mechanisms may stem from the sparse distribution of safety-critical regions within the model's architecture. Therefore, the sparsity of safety-critical neurons and ranks may act as a model-intrinsic metric for assessing the brittleness of safety alignment, complementing red teaming efforts. We hope it inspires further development of robust and reliable safety alignment algorithms that aim to integrate safety-critical regions seamlessly with utility regions for enhanced overall model performance.

2. Methodology

To identify and isolate the regions that are exclusively responsible for a model's safety behaviors, we make certain modifications to the weights of the model and observe its behavioral change in safety and utility. A natural way to modify the network weights is **neuron** removal, where we set several neurons of the weight matrices to zero. Besides, as LoRA (Hu et al., 2022) is a popular parameter-efficient fine-

[2]Our method is generally applicable to the attribution of different behaviors, but in the context of this study, the behaviors of interest are 'safety' and 'utility'.

tuning technique, we also consider **rank** removal, where we remove several ranks of the weight matrices.

For a given calibration dataset, we consider two types of behavioral changes after we modify the network weights: **output change**, where we directly monitor the change of the immediate outputs of each of the modified layers; and **loss change**, where we monitor the change of the final loss.

The section is organized as follows. In Section 2.1, we consider the three approaches on weight attribution for a general calibration dataset, and summarize them in Table 1. In Section 2.2, we illustrate how to isolate safety-critical regions that leads to significant changes in safety behavior while having minimal effect on the general capabilities of the models.

2.1. Identifying Important Neurons and Ranks

SNIP score (Lee et al., 2019) For a data instance $x = (x_{\text{prompt}}, x_{\text{response}})$, we take the loss as the conditional negative log-likelihood $\mathcal{L}(x) = -\log p(x_{\text{response}} \mid x_{\text{prompt}})$ predicted by the model. For any linear layer with weight matrix $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, we can calculate the importance score for the loss $\mathcal{L}(x)$ for each weight entry W_{ij}

$$I(W_{ij}, x) = |W_{ij} \cdot \nabla_{W_{ij}} \mathcal{L}(x)|,$$

which is the first-order Taylor approximation to the change of the loss when the weight entry W_{ij} is set to zero. In matrix form, we have

$$I(W, x) = |W \odot \nabla_W \mathcal{L}(x)|.$$

Given a calibration dataset D, we take the absolute value first and then take the average over D and obtain

$$I(W) = \mathbb{E}_{x \sim D} I(W, x) = \mathbb{E}_{x \sim D} |W \odot \nabla_W \mathcal{L}(x)|.$$

where we get an individual score for each example and aggregate over the examples following Michel et al. (2019). Intuitively, $I(W)$ measures how important each entry is for the behavior of the model on the calibration dataset D. Small $I(W)_{ij}$ indicates that setting W_{ij} to zero has negligible impact on each of the calibration data points x, and we can attribute the specific behavior of the model to the weights with large $I(W)_{ij}$.

Wanda score (Sun et al., 2024) For a calibration dataset, we store all the activations corresponding to the layer W into $X_{\text{in}} \in \mathbb{R}^{d_{\text{in}} \times n}$. We consider multiplying the weight matrix W with an element-wise binary mask M, resulting in a sparse matrix, such that the Frobenius norm of the change to the output is minimized as in Frantar & Alistarh (2023):

$$\min_{M \in \{0,1\}^{d_{\text{out}} \times d_{\text{in}}}} \|WX_{\text{in}} - (M \odot W)X_{\text{in}}\|_F^2.$$

We follow Wanda (Sun et al., 2024) to obtain an approximate solution to the objective above. Denote an all-one vector by $\mathbf{1} \in \mathbb{R}^{d_{\text{out}}}$. The importance score $I(W)$ is given by

$$I(W) = |W| \odot \left(\mathbf{1} \cdot \|X_{\text{in}}\|_2^\top\right),$$

where we take the row-wise L^2 norm to obtain $\|X_{\text{in}}\|_2 \in \mathbb{R}^{d_{\text{in}}}$. To obtain a sparse network while keeping the change to the outputs minimal, one can prune out weight neurons corresponding to the minimal score $I(W)$.

In our setting, as we are only interested in measuring the importance of each weight entry that contributes to the model's response, we mask out the prompt activations and only include the response activations in X_{in}.

ActSVD Recall that we store all the response activations before the layer W into $X_{\text{in}} \in \mathbb{R}^{d_{\text{in}} \times n}$. We seek to find a low-rank matrix $\widehat{W}$ such that the Frobenius norm of the change to the output is minimized:

$$\widehat{W} = \underset{\text{rank}\, \widehat{W} \leq r}{\arg\min} \|WX_{\text{in}} - \widehat{W}X_{\text{in}}\|_F^2.$$

This can be done by performing SVD on $WX_{\text{in}} \in \mathbb{R}^{d_{\text{out}} \times n}$:

$$USV^\top \approx WX_{\text{in}},$$

where $U \in \mathbb{R}^{d_{\text{out}} \times r}$ is the orthogonal matrix corresponding to the top r left singular vectors. The minimizer is given by

$$\widehat{W} = UU^\top W,$$

where $\Pi = UU^\top$ is the orthogonal projection onto the r most significant left singular subspace. The proof is postponed to Appendix D. Note that $\widehat{W}$ can be implemented by a LoRA update. The rank of the LoRA adaptor $\Delta W = W - \widehat{W}$ can be bounded by

$$\text{rank}(\Delta W) \leq \text{rank}(I - UU^\top) = R - r,$$

where $R = \text{rank}(W)$. With a slight abuse of terminology, we refer to these rank-1 components $(U_1 U_1^\top W,\ U_2 U_2^\top W, \dots)$ as the important *ranks* of the weight matrix W. See Appendix A.3 for a detailed discussion of the related methods.

2.2. Isolating Safety-Critical Neurons and Ranks

Assume we have two calibration different datasets: a **utility** dataset D^u and a **safety** dataset D^s. D^u contains prompts and responses that are related to general language abilities.

[3]We exclude the rank attribution + loss change combination in our paper due to several technical challenges, e.g., differentiating through an orthogonal projection matrix requires Lie algebra analysis (Schotthöfer et al., 2022).

	Neuron Attribution	Rank Attribution
Output Change	Wanda (Sun et al., 2024) $\lvert W\rvert \odot \left(\mathbf{1} \cdot \lVert X_{\text{in}}\rVert_2^\top\right)$	ActSVD $USV^\top \approx WX_{\text{in}},\ \widehat{W} = UU^\top W$
Loss Change	SNIP (Lee et al., 2019) $\mathbb{E}_{x\sim D}\lvert W \odot \nabla_W \mathcal{L}(x)\rvert$	N/A[3]
Disentanglement Method	Set difference $S(p,q) = S^s(q) - S^u(p)$	Orthogonal projection $\Delta W(r^u, r^s) = (I - \Pi^u)\Pi^s W$

Table 1. The overview of our weight attribution methods. For **neuron** attribution, we compute Wanda importance score (Sun et al., 2024) as per output change and SNIP importance score (Lee et al., 2019) as per loss change. To disentangle safety from utility, we adopt set difference to isolate the safety-critical neurons. For **rank** attribution, we compute the most important ranks via ActSVD per output change and adopt orthogonal projection to isolate the safety-critical ranks.

D^s demonstrates behavior where the requests are harmful and the responses properly decline the requests.

We seek to isolate the safety-critical regions of the network weights, where removing those regions will have a *low influence* on the model's behavior on distribution D^u but a *high influence* on D^s — effectively maintaining the model's general language abilities but compromising its safety alignment.

Isolating safety-critical neurons For the two calibration datasets, assume we obtain two scores I^u and I^s, respectively, either using SNIP or Wanda as in Section 2.1. We consider the weight neurons that score least according to I^u but score most according to I^s. We adopt *per-output comparison group* as Sun et al. (2024), which corresponds to each matrix row. Specifically, for any pair of sparsity levels $(p\%, q\%)$, we define the top-$p\%$ important neurons $S^u(p)$ for utility as the neurons whose important utility score $I^u_{i,j}$ ranks top $p\%$ among the i-th *row* of I^u.

$$S^u(p) = \{(i,j) \mid I^u_{i,j} \text{ is the top } p\% \text{ of } I^u_i\}.$$

Similarly, we define the top-$q\%$ important neurons $S^s(q)$ for safety as

$$S^s(q) = \{(i,j) \mid I^s_{i,j} \text{ is the top } q\% \text{ of } I^s_i\}.$$

Then, the isolated neurons $S(p,q)$ is defined as the set difference between $S^s(q)$ and $S^u(p)$:

$$S(p,q) = S^s(q) - S^u(p).$$

Isolating safety-critical ranks Let $R = \text{rank}(W)$. For the two calibration datasets D^u and D^s, for any pair of integer (r^u, r^s), we can obtain the projection matrices Π^u and Π^s as stated in Section 2.1, where

$$\Pi^u = U^u(U^u)^\top, \text{rank}(\Pi^u) = R - r^u,$$

and

$$\Pi^s = U^s(U^s)^\top, \text{rank}(\Pi^s) = R - r^s.$$

Multiplying the weight matrix by Π^u removes the least important r^u ranks for D^u, while multiplying the weight matrix by Π^s removes the least important r^s ranks for D^s. To isolate the safety-critical ranks, we consider the matrix

$$\Delta W(r^u, r^s) = (I - \Pi^u)\Pi^s W.$$

Removal of $\Delta W(r^u, r^s)$ essentially removes the important ranks of the safety behavior that are *orthogonal* to the important ranks of the utility behavior. The modified weight matrix is $\widetilde{W} = W - \Delta W(r^u, r^s)$, which can be implemented by a LoRA update (Hu et al., 2022) with

$$\text{rank}(\Delta W(r^u, r^s)) \leq \min(r^u, R - r^s).$$

3. Experimental Setup

3.1. Models, Datasets, and Evaluation Metrics

Models Our experiments use `Llama2-7B-chat` and `Llama2-13B-chat` (Touvron et al., 2023b). We select them for their publicly accessible weights and their extensive safety tuning process.

Datasets To identify safety-critical regions in the model, we prepare two types of datasets: the safety dataset, for attributing safety-related behaviors, and the utility dataset, for attributing utility-related behaviors. Each dataset is structured in a (prompt, response) format. More details about these datasets are provided in Appendix B.

The safety dataset is compiled using harmful instructions from `AdvBench` (Zou et al., 2023a). We divide `AdvBench` into $\texttt{AdvBench}_{\texttt{eval}}$ (100 instructions for evaluation) and $\texttt{AdvBench}_{\texttt{attr}}$ (420 instructions for attribution). We prompt `Llama2-7B-chat` with $\texttt{AdvBench}_{\texttt{attr}}$, collecting responses that refrain from following harmful instructions. As noted by Zou et al. (2023b), the *judgement* segments in the model's responses (e.g.,

	$ASR_{Vanilla}$	$ASR_{Adv\text{-}Suffix}$	$ASR_{Adv\text{-}Decoding}$
Sample Times	1	1	5
System Prompt	☑	✖	✖
`[INST]`, `[/INST]` wrapper	☑	☑	✖
Adversarial Suffix	✖	☑	✖

Table 2. The differences between three types of ASR in our safety evaluation. $ASR_{Vanilla}$ captures the model behavior under standard usage, while $ASR_{Adv\text{-}Suffix}$ and $ASR_{Adv\text{-}Decoding}$ captures behaviors against adversaries.

"Sure," "I am sorry") significantly impact the nature of subsequent responses. We thus create two variants: safety-full (entire response) and safety-short (judgement segment only).

For the utility dataset, we filter out safety-related (prompt, response) pairs using sensitive phrase matching (Qi et al., 2024b) from `Alpaca-Cleaned`[4], a refined version of the Alpaca dataset (Taori et al., 2023).

Measuring utility Following Sun et al. (2024), we measure the model's utility by reporting its averaged zero-shot accuracy of six tasks from EleutherAI LM Harness (Gao et al., 2023): BoolQ (Clark et al., 2019a), RTE (Wang et al., 2019), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2021), ARC Challenge (Clark et al., 2018), and OpenbookQA (Mihaylov et al., 2018).

Measuring safety We measure the model's safety by evaluating its attack success rate (ASR) in response to harmful instructions. Specifically, we prompt the model using $\texttt{AdvBench}_{\texttt{eval}}$, the first 100 prompts from `AdvBench`, and collect its responses. Following Zou et al. (2023b), we consider an attack as successful if the model's response lacks key patterns indicative of instruction rejection. The ASR is then computed as the ratio of successfully attacked prompts to the total number of prompts evaluated.

Our safety evaluation considers three use cases: the ASR under standard, non-malicious conditions ($ASR_{Vanilla}$), and the ASR under two malicious settings – $ASR_{Adv\text{-}Decoding}$ (Huang et al., 2024), where the attacker manipulates the decoding process, and $ASR_{Adv\text{-}Suffix}$ (Zou et al., 2023b), where the attacker optimizes to find adversarial suffixes. Differences in these metrics are detailed in Table 2. Due to the high computational cost associated with calculating adversarial suffixes, we precompute several suffixes, and use the three best-performed ones in our evaluation. Note that we only include the system prompt when calculating $ASR_{Vanilla}$. More details are provided in Appendix B.

[4] https://github.com/gururise/AlpacaDataCleaned

3.2. Variants to Identify Safety-Critical Neurons

We conduct experiments to identify safety-critical neurons using the following methods:

- **SNIP (top)**: we regard neurons that receive top-$p\%$ SNIP scores (Section 2.1) on safety data as safety-critical. We choose $p = 0.01$.
- **Wanda (top)**: we regard neurons that receive top-$p\%$ Wanda scores (Section 2.1) on safety data as safety-critical. We choose $p = 0.01$.
- **SNIP[5] with set difference**: we identify safety-critical neurons by focusing on those with top $q\%$ scores in the safety data, which are not included in the top $p\%$ scoring neurons according to the utility data (Section 2.2). We do a grid search for parameters p and q, with their values ranging from 0.1 to 90.
- **Probing**: we also compare our approach with probing (Hewitt & Liang, 2019), a common method for attributing behaviors of LLMs to their internal components. Following standard probing practices (Clark et al., 2019b; Campbell et al., 2023; Li et al., 2023a), we feed the model both harmful and harmless instructions, collect activation outputs from each attention head, and then train a linear classifier for each head to differentiate these activations. Attention heads with the highest accuracy on the evaluation set are identified as safety-critical. Appendix B provides more details.

3.3. Variants to Identify Safety-Critical Ranks

We conduct experiments to identify safety-critical ranks using the following methods:

- **ActSVD (top)**: we regard the top-r ranks identified as most safety-related by ActSVD (Section 2.1) as safety-critical. We choose $r = 1$.
- **ActSVD with orthogonal projection**: we identify the safety-critical ranks via orthogonal projection between the utility projection matrix Π^u and the safety projection matrix Π^s obtained from ActSVD (Section 2.2). We do a grid search for r^u and r^s between 50 and 4000.

4. Experimental Results

This section presents our findings on both neuron and rank levels. In Section 4.1, we demonstrate that our set difference and orthogonal projection outperform other methods in isolating safety-critical neurons or ranks. In Section 4.2, we show that the safety of the model can be enhanced by removing the least important safety neurons or ranks. Then in Section 4.3, we analyze the overlap between safety and utility neurons and ranks, where we observe less overlap in MLP layers than self attention layers. Finally, in Section 4.4

[5] We only use SNIP for set difference because during our early experiments, we find SNIP performs slightly better than Wanda.

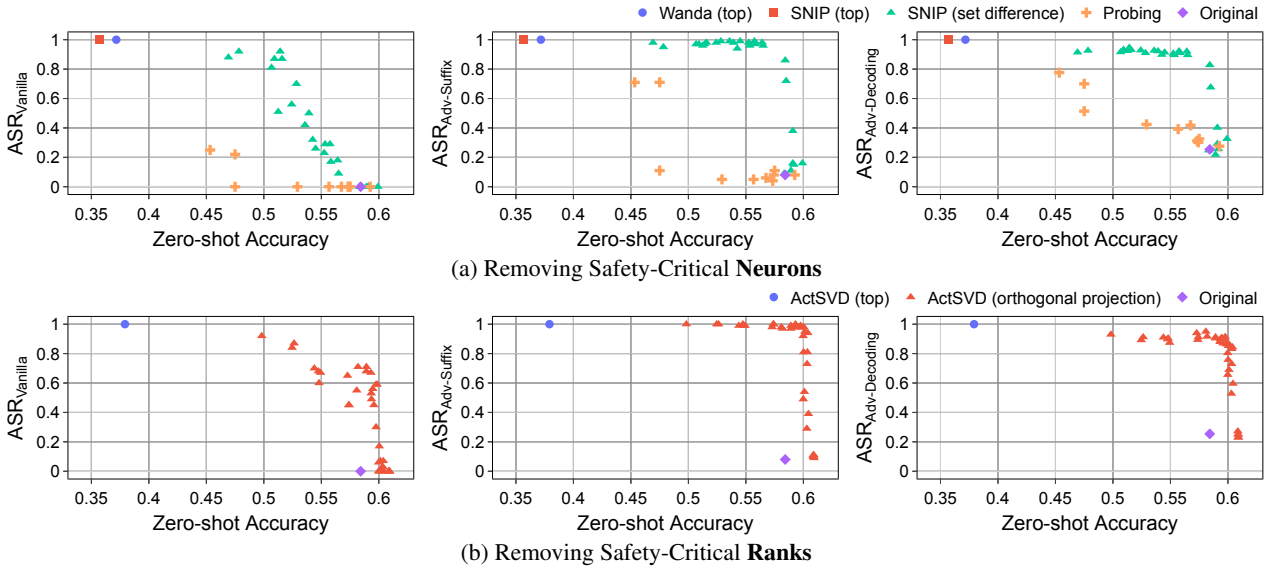


Figure 2. ASR and accuracy after removing safety-critical regions in `Llama2-7B-chat` identified by: **(a)** Different methods in Section 3.2 with **sparsity constraint** $< 3\%$. **(b)** Different methods in Section 3.3 with **ranks of the weight updates** ($\text{rank}(\Delta W)$) **less than 100** (out of 4096). Among all methods, disentangling safety from utility (set difference for neurons and orthogonal projection for ranks) mostly effectively identify the safety-critical regions, with safety severely compromised while utility retains. We obverse similar results on `Llama2-13B-chat` (see Figure 5a in Appendix C.1).

we examine the potential of freezing safety-critical neurons to counter fine-tuning attacks and explore how fine-tuning may circumvent safety mechanisms.

4.1. Disentangling Safety and Utility is Vital for Identifying Safety-Critical Regions

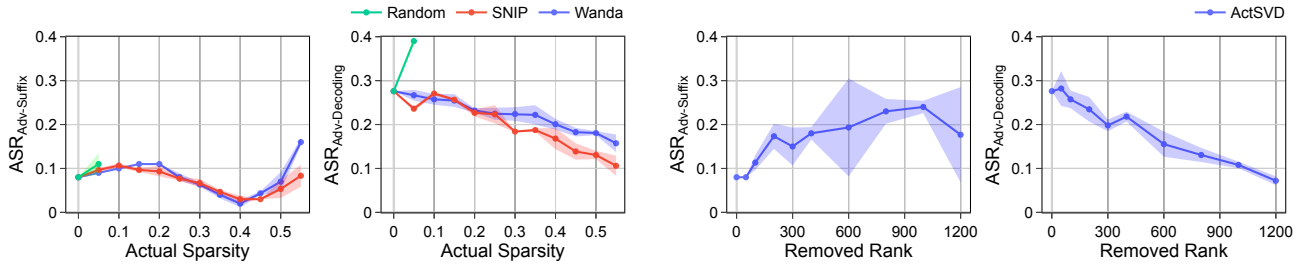
We experiment with different methods to identify safety-critical neurons and ranks outlined in Section 3.2 and Section 3.3, and summarize our findings as below.

Safety-critical regions are sparse and can be effectively isolated via set difference or orthogonal projection. We isolate neurons contributing to safety from those contributing to utility, by applying the set difference method described in Section 2.2 to SNIP. Figure 2a presents the Pareto front resulting from set difference-based pruning: We observe that removing less than 3% of neurons pushes ASR in all three scenarios close to 1, while maintaining an average zero-shot accuracy above 0.5. Similarly, Figure 2b shows results for removing safety ranks orthogonal to utility ranks: $\Delta W(r^u, r^s) = (I - \Pi^u)\Pi^s W$. Notably, an update of just 2.5% (less than 100) of the total 4096 ranks significantly increases the model's ASR, while preserving its zero-shot accuracy. For example, when we remove the orthogonally-projected top-6 safety ranks while keeping the top-96 utility ranks untouched, we get $0.71, 0.97, 0.91$ in $\text{ASR}_{\text{Vanilla}}$, $\text{ASR}_{\text{Adv-Suffix}}$ and $\text{ASR}_{\text{Adv-Decoding}}$ respectively and 0.58 zero-shot accuracy. These findings suggest that regions critical for safety are relatively *sparse* within the weight matrices, evident at both the neuron and rank levels.

Pruning merely less than 1% neurons makes the model vulnerable in adversarial cases. We also observe from Figure 2 (middle and right) that models tend to be more fragile in adversarial scenarios, as indicated by the non-dropping accuracy when $\text{ASR}_{\text{Adv-Suffix}}$ and $\text{ASR}_{\text{Adv-Decoding}}$ reach to 1. Interestingly, if we focus solely on adversarial cases, only pruning less than 1% of neurons can significantly compromise the model's safety while still keeping its accuracy above 0.53, as shown in Table 6 in Appendix C.3.

Pruning top safety neurons or ranks severely compromises utility. Removing neurons with the highest safety importance scores, either calculated with Wanda or SNIP, also leads to a complete loss of safety in the `Llama2-7B-chat` model. However, there is also a drastic decrease in the model's utility, with its average accuracy dropping to about 0.35, significantly lower than its original accuracy of 0.58. Likewise, removing even just the top-1 rank (out of 4096) critical for safety causes the model's accuracy to drop to 0.38. Similar results are also observed in `Llama2-13B-chat` (Figure 5a in Appendix C.1).

These observations support our rationale for isolating safety from utility: Regions primarily contributing to safety behaviors in aligned models may also be crucial for its general utility. Consequently, removing these regions can impair the model's ability to generate grammatically correct content, which in turn undermines its safety mechanisms.



(a) Removing the least Safety-Relevant **Neurons**

(b) Removing the least Safety-Relevant **Ranks**

Figure 3. Impact on ASR under adversaries in `Llama2-7B-chat` when: (a) Removing neurons with *the lowest* safety importance scores and (b) Removing *the least* safety-relevant ranks identified by ActSVD. Both the importance score and safety-relevant ranks are calculated on the safety-short dataset. All plotted points have reasonable utility with accuracy > 0.5. The exclusion of neurons and ranks deemed least critical for safety results in improved robustness against adversarial decoding attacks. Potential reason for the variation in $\text{ASR}_{\text{Adv-Suffix}}$ is the adversarial suffixes may not directly transfer to the modified model. See Figure 6 for the results on `Llama2-13B-chat`.

Attention head probing scores cannot isolate safety-critical neurons. Our probing results in Appendix C.4 suggest that activations of individual attention heads are predictive of identifying harmful versus harmless instructions, with over half achieving more than 0.95 probing accuracy. Based on the obtained score of each attention head, we prune the top-k scored (out of 1024) attention heads from `Llama2-7B-chat`, with k ranging from 10 to 300. However, as shown in Figure 2a, our set difference approach outperforms the probing method consistently, yielding higher ASR at the same level of accuracy. Similar results are observed on `Llama2-13B-chat` (see Figure 5a). This highlights the need for a disentanglement method, as achieving high harmful versus harmless prediction accuracy does not necessarily mean the top predictive heads are solely responsible for generating safety responses. Besides, these results also imply the need to focus on the MLP layers and a finer granularity like neurons or ranks.

4.2. Enhancing Safety by Eliminating Regions with Minimal Safety Relevance

Pruning least safety-relevant neurons improves safety. Thinking from the opposite, it's reasonable to hypothesize that the neurons with the lowest safety importance scores could be detrimental for safety. Consequently, eliminating these neurons could potentially enhance the overall safety of the model. To verify this, we conduct an ablation study to prune weights with the lowest SNIP and Wanda scores at various sparsity levels ($s\%$), and report randomly removing $s\%$ of neurons as a baseline for comparison. Particularly, we only report the results for pruned models maintaining reasonable utility, defined by an average accuracy above 0.5. As shown in Figure 3a, random pruning strategy significantly reduces model accuracy, which falls below 0.5 after pruning merely 10% of the weights; it also leads to a noticeable decline in the model's safety. In contrast, when pruning is guided by the lowest safety importance score, the model's accuracy remains largely stable (i.e., > 0.5). Furthermore, pruning neurons with the lowest safety scores even slightly enhances the model's safety, against both decoding manipulation and adversarial suffixes. This indicates that neurons identified as least important for safety may undermine the model's safety, thereby validating the effectiveness of our method in calculating safety importance scores.

Removing least safety-relevant ranks improves safety. Similarly, we also consider removing the least safety ranks from the model using ActSVD. Specifically, we remove least $(R - r)$ safety ranks by using $\widehat{W} = UU^\top W = \sum_{i=1}^{r} U_i U_i^\top W$ to approximate W. In accordance to our findings at the neuron level, as we increase the removed rank, we observe a decrease in the model's $\text{ASR}_{\text{Adv-Decoding}}$ in Figure 3b. This also echoes the recent findings (Sharma et al., 2023) where removing high-order components improves the model's reasoning performance. However, the model's $\text{ASR}_{\text{Adv-Suffix}}$ exhibits considerable variation, which could potentially be due to that the adversarial suffixes found using Zou et al. (2023b) on the original model cannot directly transfer to the modified model.

4.3. MLP Layers Appear to Encode More Differentiated Behaviors

In Section 4.1, we find that removing high-safety-score regions also compromises utility, indicating a possible entanglement of safety and utility regions. We validate this at both neuron and rank levels.

Neuron-level Jaccard index: We calculate the layer-wise Jaccard index, $J(A, B) = |A \cap B|/|A \cup B|$, to quantify the overlap between top $p\%$ utility neurons and top $q\%$ safety neurons. Figure 4a shows Jaccard indices across all transformer blocks and layers in `Llama2-7B-chat`, using SNIP importance scores with top percentages $p\%$

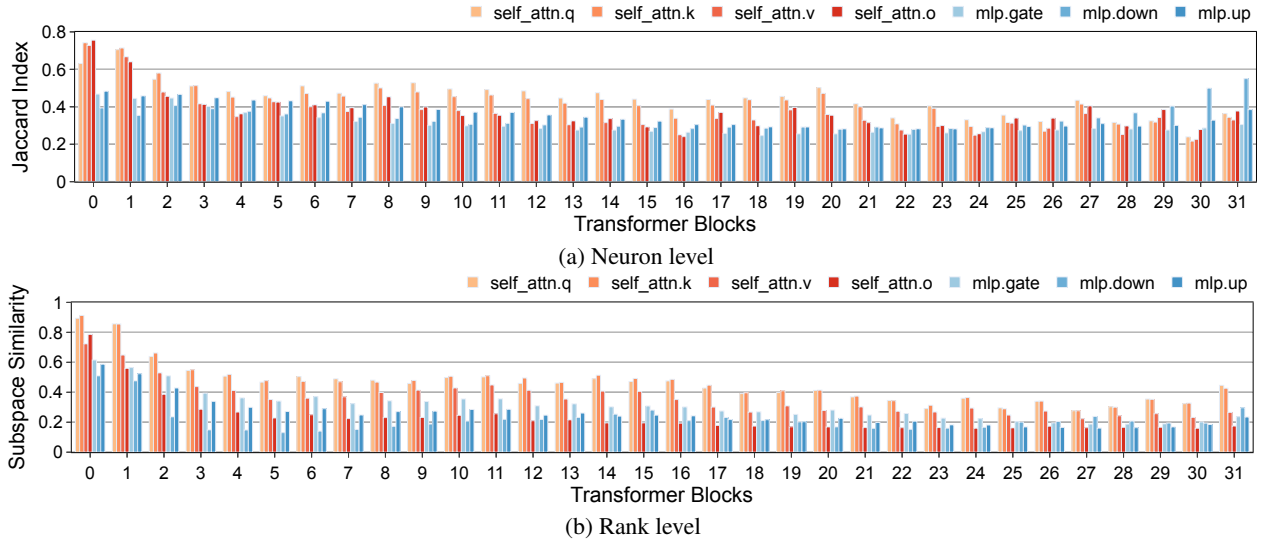


Figure 4. Safety and utility overlapping analysis of each layer inside `Llama2-7B-chat` using: (a) Jaccard Index between top 5% safety neurons and top 5% utility neurons and (b) Subspace similarity between rank-100 utility matrix U^u and rank-100 safety matrix U^s. **Lower** Jaccard index/subspace similarity means lower overlapping of utility and safety, namely utility and safety behaviors are **more differentiated**. MLP layers appear to encode more differentiated behaviors.

and $q\%$ at 5%[6]. The observed spikes in Jaccard indices indicate large overlaps between safety and utility neurons within certain layers of the model. Notably, MLP layers exhibit lower Jaccard indices compared to attention layers, suggesting that utility or safety-related knowledge is more differentiated in MLP layers within language models (Dai et al., 2022).

Rank-level subspace similarity: Similarly, we also find that MLP layers appear to encode more differentiated behaviors for safety and utility, from the rank perspective. Specifically, we report[7] the subspace similarity $\phi(U^u, U^s)$ between rank-100 U^u and rank-100 U^s as defined in Hu et al. (2022, Appendix G) in Figure 4b, where

$$\phi(U^u, U^s) = \frac{\|(U^u)^\top U^s\|_F^2}{\min(\text{rank}(U^u), \text{rank}(U^s))}.$$

As shown, the left singular matrices U^u and U^s exhibit a lower subspace similarity (i.e., utility and safety behaviors are more differentiated) in MLP layers, corroborating our findings at the neuron level.

4.4. Freezing Safety-Critical Neurons Does Not Stop Fine-Tuning Attacks

Finally, we explore the implications of the identified neurons on the fine-tuning attacks, which demonstrate that fine-tuning an aligned model, even with harmless data, can unexpectedly weaken its safety measures (Qi et al., 2024b; Yang et al., 2023; Zhan et al., 2023). This issue is particularly concerning given the increasing availability of model fine-tuning APIs from major vendors like OpenAI[8].

% of Frozen Weights	$\text{ASR}_{\text{Vanilla}}$		
	$n = 10$	$n = 50$	$n = 100$
0 (original attack)	0.53	0.91	0.94
9.61	0.52	0.92	0.94
19.22	0.40	0.91	0.94
28.83	0.38	0.91	0.94
48.05	0.34	0.91	0.94
67.27	0.23	0.85	0.89

Table 3. $\text{ASR}_{\text{Vanilla}}$ of the model under fine-tuning with varying fractions of safety-critical neurons frozen. n represents the number of Alpaca examples used for fine-tuning. Freezing safety-critical neurons is insufficient to thwart fine-tuning attacks.

We explore whether the identified safety-critical neurons could mitigate the fine-tuning attack[9]. Following the experimental setup in Qi et al. (2024b), we fine-tune `Llama2-7B-chat` with varying numbers of examples (n) from the Alpaca dataset (Taori et al., 2023). During fine-tuning, we freeze the top-$q\%$ of safety neurons and observe their effect on preserving safety. As shown in Table 3, effective counteraction of the attack occurs only with $n = 10$ and freezing over 50% of neurons. This observation aligns with Lee et al. (2024)'s hypothesis that fine-tuning attacks may create alternative pathways in the original model. Given that safety-critical neurons are sparse, these new routes could

[6]We choose top 5% because the optimal selections of $(p\%, q\%)$ presented in Figure 2 is around 5% (see Appendix C.3).

[7]We choose top 100 ranks as the optimal (r^u, r^s) corresponds to top 100 safety and utility ranks (see Appendix C.3).

[8]`https://platform.openai.com/finetune`

[9]We only explore this for neurons, as the "freezing" operation at rank level cannot be easily achieved using U^u and U^s.

bypass the existing safety mechanisms easily, and therefore we need more robust defenses against fine-tuning attacks.

5. Limitations & Future Work

We identify areas for potential future research and limitations of this work. First, there are limited publicly accessible, strong safety-aligned models, which constrains our experiments to the Llama2-chat models. Other aligned models, trained with different datasets and strategies, might demonstrate varying behaviors under our methodology.

Second, we found that standard attention head probing does not effectively localize safety-critical neurons. Our findings that MLP layers may exhibit better localization for safety knowledge, also suggest that future probing research could explore MLP layers in more depth. This exploration could also examine the potential integration of these methods with our pipeline to enhance safety attribution effectiveness.

Our study proposes initial, yet promising, strategies for improving safety robustness, which could be explored further: (1) pruning regions least important for safety (or potentially explicitly harmful) could improve safety robustness; (2) making safety-critical regions difficult to isolate may be an exciting new direction in building inherently safer models.

6. Conclusion

In this study, we introduce a pipeline for identifying safety-critical regions (neurons and ranks) in LLMs, which effectively disentangles the regions critical for safety and those vital for utility. Our experiments with Llama2-chat models demonstrate that safety-critical regions are notably sparse in aligned LLMs, accounting for about 3% at the weight level and 2.5% at the rank level. Despite their sparsity, these regions are crucial for the integrity of the model's safety mechanisms, as removing them destroys the model's safety with utility retained. This sparsity may explain the observed brittleness in safety alignment in current LLMs, and could serve as a model-intrinsic metric for assessing the brittleness of safety alignment in future models, thereby complementing red teaming efforts. And our work suggests potentially important future directions for improving the robustness and safety of models overall.

Acknowledgements

We express our gratitude to Vikash Sehwag, Chiyuan Zhang, Yi Zeng, Ruoxi Jia, Lucy He, Kaifeng Lyu, and the Princeton LLM Alignment reading group for providing helpful feedback. Boyi Wei and Tinghao Xie are supported by the Francis Robbins Upton Fellowship, Yangsibo Huang is supported by the Wallace Memorial Fellowship, and Xiangyu Qi is supported by Gordon Y. S. Wu Fellowship. Prateek Mittal acknowledges the support by NSF grants CNS-1553437 and CNS-1704105, the ARL's Army Artificial Intelligence Innovation Institute (A2I2), the Office of Naval Research Young Investigator Award, the Army Research Office Young Investigator Prize, Schmidt DataX award, and Princeton E-affiliates Award. Mengdi Wang acknowledges the support by NSF IIS-2107304, NSF CPS-2312093, ONR 1006977, and Genmab. This research is also supported by the Center for AI Safety Compute Cluster. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the sponsors.

Contribution Statement

This project was a team effort. The contributions of each junior author are detailed below:

Idea formulation and preliminary exploration: The project's core ideas, specifically linking safe behaviors to certain model regions and isolating safety from utility, were developed by Boyi, Kaixuan, and Yangsibo. Yangsibo and Boyi implemented Wanda and set difference methods based on discussions with Mengzhou, and ran preliminary experiments. Kaixuan implemented SNIP, ActSVD, and orthogonal projection methods, and ran preliminary experiments.

Literature survey: The literature review was conducted by team members in their respective areas of expertise. Tinghao focused on alignment and jailbreaks; Yangsibo explored task attribution; Kaixuan reviewed low-rank modifications, and Mengzhou surveyed pruning techniques.

Evaluation: Boyi led the charge on key experiments and the creation of visuals, while the entire team played a part in the evaluation process. Yangsibo and Tinghao prepared the attribution data and set up the evaluation pipeline. Xiangyu collaborated with Boyi on reporting results for the attention head probing baseline. Boyi and Kaixuan investigated Jaccard index and subspace similarity to analyze the overlapping between safety and utility. Yangsibo and Tinghao studied the effect of freezing safety-critical regions during the fine-tuning.

Writing: The initial structure and primary drafting of the manuscript were led by Yangsibo, Boyi, and Kaixuan. The rest of the team members contributed by editing and providing valuable feedback on the manuscript.

Impact Statement

Dual-use Risk. Our work, like other safety and security research, aims to make models safer in the long run by identifying short-term weaknesses. We hope our work will spur additional researching robust safety mechanisms that are not so sparse, easy to isolate, and easy to remove.

That being said, with any safety and security research there is some risk that adversaries will use our work to remove safety guardrails. We believe the benefit of releasing our work and engaging in this study outweighs these potential risks for three reasons.

First, we perform experiments on models (Llama2-chat family) that already has a base model available without any safety guardrails, so there is no marginal increased risk. Second, by assessing the brittleness of safety guardrails in our work, we can encourage stronger guardrails to be developed that are more difficult for an attacker to isolate and remove. Our work may help explain existing results demonstrating that safety guardrails can be removed via fine-tuning, while identifying potential pathways for improved defenses. Third, our work does not significantly decrease the cost of jailbreaking a model beyond alternative strategies. Models can already be jailbroken relatively cheaply with fine-tuning. Instead, our focus is on analysis and understanding of the brittleness of these safety mechanisms so that future work can reduce the risk of jailbreaking in open models.

Overall, we hope that our work will improve the state of AI safety, particularly in open models, by providing key analysis and information.

Safety and harm definitions. We generally follow existing standard benchmarks and protocols for assessment of safety and harm, but these may not cover all definitions of safety and harm. Further work can be done to expand analysis to a wider range of settings and we encourage additional work in the space of definitions and evaluation that are beyond the scope of this work.

References

Adi, Y., Kermany, E., Belinkov, Y., Lavi, O., and Goldberg, Y. Fine-grained Analysis of Sentence Embeddings Using Auxiliary Prediction Tasks. In *ICLR*, 2016.

Albert, A. Jailbreak Chat, 2023. URL https://www.jailbreakchat.com/.

Ali, S., Abuhmed, T., El-Sappagh, S., Muhammad, K., Alonso-Moral, J. M., Confalonieri, R., Guidotti, R., Del Ser, J., Díaz-Rodríguez, N., and Herrera, F. Explainable Artificial Intelligence (XAI): What We Know and What Is Left to Attain Trustworthy Artificial Intelligence. *Information Fusion*, 2023.

Anthropic. Claude, 2023a. URL https://claude.ai/.

Anthropic. Claude 2, 2023b. URL https://www.anthropic.com/news/claude-2.

Antverg, O. and Belinkov, Y. On the Pitfalls of Analyzing Individual Neurons in Language Models. In *ICLR*, 2021.

Ashkboos, S., Croci, M. L., Nascimento, M. G. d., Hoefler, T., and Hensman, J. SliceGPT: Compress Large Language Models by Deleting Rows and Columns. *arXiv preprint arXiv:2401.15024*, 2024.

Askell, A., Bai, Y., Chen, A., Drain, D., Ganguli, D., Henighan, T., Jones, A., Joseph, N., Mann, B., DasSarma, N., et al. A General Language Assistant as a Laboratory for Alignment. *arXiv preprint arXiv:2112.00861*, 2021.

Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R., and Samek, W. On Pixel-Wise Explanations for Non-linear Classifier Decisions by Layer-Wise Relevance Propagation. *PloS one*, 2015.

Bai, Y., Jones, A., Ndousse, K., Askell, A., Chen, A., DasSarma, N., Drain, D., Fort, S., Ganguli, D., Henighan, T., et al. Training a Helpful and Harmless Assistant With Reinforcement Learning From Human Feedback. *arXiv preprint arXiv:2204.05862*, 2022a.

Bai, Y., Kadavath, S., Kundu, S., Askell, A., Kernion, J., Jones, A., Chen, A., Goldie, A., Mirhoseini, A., McKinnon, C., et al. Constitutional AI: Harmlessness From AI Feedback. *arXiv preprint arXiv:2212.08073*, 2022b.

Belinkov, Y. Probing Classifiers: Promises, Shortcomings, and Advances. *Computational Linguistics*, 2022.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language Models are Few-Shot Learners. In *NeurIPS*, 2020.

Burkart, N. and Huber, M. F. A Survey on the Explainability of Supervised Machine Learning. *Journal of Artificial Intelligence Research*, 2021.

Burns, C., Ye, H., Klein, D., and Steinhardt, J. Discovering Latent Knowledge in Language Models Without Supervision. In *ICLR*, 2023.

Campbell, J., Guo, P., and Ren, R. Localizing Lying in Llama: Understanding Instructed Dishonesty on True-False Questions Through Prompting, Probing, and Patching. In *Socially Responsible Language Modelling Research*, 2023.

Cao, S., Sanh, V., and Rush, A. M. Low-Complexity Probing via Finding Subnetworks. In *NAACL*, 2021.

Carlini, N., Nasr, M., Choquette-Choo, C. A., Jagielski, M., Gao, I., Awadalla, A., Koh, P. W., Ippolito, D., Lee, K., Tramer, F., et al. Are Aligned Neural Networks Adversarially Aligned? In *NeurIPS*, 2023.

Chen, T., Frankle, J., Chang, S., Liu, S., Zhang, Y., Wang, Z., and Carbin, M. The Lottery Ticket Hypothesis for Pre-trained BERT Networks. *NeurIPS*, 2020.

Chen, T., Ding, T., Yadav, B., Zharkov, I., and Liang, L. LoRAShear: Efficient Large Language Model Structured Pruning and Knowledge Recovery. *arXiv preprint arXiv:2310.18356*, 2023.

Chiang, W.-L., Li, Z., Lin, Z., Sheng, Y., Wu, Z., Zhang, H., Zheng, L., Zhuang, S., Zhuang, Y., Gonzalez, J. E., Stoica, I., and Xing, E. P. Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality, March 2023. URL https://lmsys.org/blog/2023-03-30-vicuna/.

Clark, C., Lee, K., Chang, M.-W., Kwiatkowski, T., Collins, M., and Toutanova, K. BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions. In *NAACL*, 2019a.

Clark, K., Khandelwal, U., Levy, O., and Manning, C. D. What does BERT look at? An Analysis of BERT's Attention. In *ACL*, 2019b.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Conneau, A., Kruszewski, G., Lample, G., Barrault, L., and Baroni, M. What You Can Cram Into a Single Vector: Probing Sentence Embeddings for Linguistic Properties. In *ACL*, 2018.

Dai, D., Dong, L., Hao, Y., Sui, Z., Chang, B., and Wei, F. Knowledge Neurons in Pretrained Transformers. In *ACL*, 2022.

Dai, J., Pan, X., Sun, R., Ji, J., Xu, X., Liu, M., Wang, Y., and Yang, Y. Safe RLHF: Safe Reinforcement Learning from Human Feedback. In *ICLR*, 2024.

Dalvi, F., Durrani, N., Sajjad, H., Belinkov, Y., Bau, A., and Glass, J. What Is One Grain of Sand in the Desert? Analyzing Individual Neurons in Deep NLP Models. In *AAAI*, 2019.

Eckart, C. and Young, G. The Approximation of One Matrix by Another of Lower Rank. *Psychometrika*, 1(3):211–218, 1936.

Frankle, J. and Carbin, M. The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks. In *ICLR*, 2018.

Frantar, E. and Alistarh, D. SparseGPT: Massive Language Models Can Be Accurately Pruned in One-Shot. In *ICML*, 2023.

Gao, L., Tow, J., Abbasi, B., Biderman, S., Black, S., DiPofi, A., Foster, C., Golding, L., Hsu, J., Le Noac'h, A., Li, H., McDonell, K., Muennighoff, N., Ociepa, C., Phang, J., Reynolds, L., Schoelkopf, H., Skowron, A., Sutawika, L., Tang, E., Thite, A., Wang, B., Wang, K., and Zou, A. A Framework for Few-shot Language Model Evaluation, 2023. URL https://zenodo.org/records/10256836.

Geva, M., Bastings, J., Filippova, K., and Globerson, A. Dissecting Recall of Factual Associations in Auto-Regressive Language Models. In *EMNLP*, 2023.

Guo, D., Rush, A. M., and Kim, Y. Parameter-efficient transfer learning with diff pruning. In *ACL*, 2021.

Gurnee, W., Horsley, T., Guo, Z. C., Kheirkhah, T. R., Sun, Q., Hathaway, W., Nanda, N., and Bertsimas, D. Universal Neurons in GPT2 Language Models. *arXiv preprint arXiv:2401.12181*, 2024.

Han, S., Mao, H., and Dally, W. J. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. In *ICLR*, 2016.

He, Y., Zhang, X., and Sun, J. Channel Pruning for Accelerating Very Deep Neural Networks. In *ICCV*, 2017.

Hewitt, J. and Liang, P. Designing and Interpreting Probes With Control Tasks. In *EMNLP*, 2019.

Hsu, Y.-C., Hua, T., Chang, S., Lou, Q., Shen, Y., and Jin, H. Language Model Compression With Weighted Low-Rank Factorization. In *ICLR*, 2021.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. LoRA: Low-Rank Adaptation of Large Language Models. In *ICLR*, 2022.

Huang, Y., Gupta, S., Xia, M., Li, K., and Chen, D. Catastrophic Jailbreak of Open-source LLMs via Exploiting Generation. In *ICLR*, 2024.

Jain, S., Kirk, R., Lubana, E. S., Dick, R. P., Tanaka, H., Grefenstette, E., Rocktäschel, T., and Krueger, D. S. Mechanistically Analyzing the Effects of Fine-Tuning on Procedurally Defined Tasks. *arXiv preprint arXiv:2311.12786*, 2023.

Jones, E., Dragan, A., Raghunathan, A., and Steinhardt, J. Automatically Auditing Large Language Models via Discrete Optimization. In *ICML*, 2023.

Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., Gonzalez, J., Zhang, H., and Stoica, I. Efficient Memory Management for Large Language Model Serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626, 2023.

Lagunas, F., Charlaix, E., Sanh, V., and Rush, A. M. Block Pruning For Faster Transformers. In *EMNLP*, 2021.

Lan, M. and Barez, F. Locating Cross-Task Sequence Continuation Circuits in Transformers. *arXiv preprint arXiv:2311.04131*, 2023.

Lee, A., Bai, X., Pres, I., Wattenberg, M., Kummerfeld, J. K., and Mihalcea, R. A Mechanistic Understanding of Alignment Algorithms: A Case Study on DPO and Toxicity. *arXiv preprint arXiv:2401.01967*, 2024.

Lee, H., Phatale, S., Mansoor, H., Lu, K., Mesnard, T., Bishop, C., Carbune, V., and Rastogi, A. RLAIF: Scaling Reinforcement Learning from Human Feedback with AI Feedback. *arXiv preprint arXiv:2309.00267*, 2023.

Lee, N., Ajanthan, T., and Torr, P. SNIP: Single-shot Network Pruning based on Connection Sensitivity. In *ICLR*, 2019.

Li, H., Kadav, A., Durdanovic, I., Samet, H., and Graf, H. P. Pruning Filters for Efficient ConvNets. In *ICLR*, 2017.

Li, K., Patel, O., Viégas, F., Pfister, H., and Wattenberg, M. Inference-Time Intervention: Eliciting Truthful Answers from a Language Model. In *NeurIPS*, 2023a.

Li, Y., Yu, Y., Zhang, Q., Liang, C., He, P., Chen, W., and Zhao, T. LoSparse: Structured Compression of Large Language Models based on Low-Rank and Sparse Approximation. In *ICML*, 2023b.

Li, Y., Wei, F., Zhao, J., Zhang, C., and Zhang, H. RAIN: Your Language Models Can Align Themselves without Finetuning. In *ICLR*, 2024.

Liu, X., Xu, N., Chen, M., and Xiao, C. AutoDAN: Generating Stealthy Jailbreak Prompts on Aligned Large Language Models. In *ICLR*, 2024.

Liu, Y., Deng, G., Xu, Z., Li, Y., Zheng, Y., Zhang, Y., Zhao, L., Zhang, T., and Liu, Y. Jailbreaking ChatGPT via Prompt Engineering: An Empirical Study. *arXiv preprint arXiv:2305.13860*, 2023.

Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., and Zhang, C. Learning Efficient Convolutional Networks through Network Slimming. In *ICCV*, 2017.

Lubana, E. S., Bigelow, E. J., Dick, R. P., Krueger, D., and Tanaka, H. Mechanistic Mode Connectivity. In *ICML*, 2023.

Lundberg, S. M. and Lee, S.-I. A Unified Approach to Interpreting Model Predictions. In *NeurIPS*, 2017.

Luo, J.-H., Wu, J., and Lin, W. ThiNet: A Filter Level Pruning Method for Deep Neural Network Compression. In *ICCV*, 2017.

Ma, X., Fang, G., and Wang, X. LLM-Pruner: On the Structural Pruning of Large Language Models. In *NeurIPS*, 2023.

Madsen, A., Reddy, S., and Chandar, S. Post-hoc Interpretability for Neural NLP: A Survey. *ACM Computing Surveys*, 2022.

Maini, P., Mozer, M. C., Sedghi, H., Lipton, Z. C., Kolter, J. Z., and Zhang, C. Can Neural Network Memorization Be Localized? In *ICML*, 2023.

Mehrotra, A., Zampetakis, M., Kassianik, P., Nelson, B., Anderson, H., Singer, Y., and Karbasi, A. Tree of Attacks: Jailbreaking Black-Box LLMs Automatically. *arXiv preprint arXiv:2312.02119*, 2023.

Meng, K., Bau, D., Andonian, A., and Belinkov, Y. Locating and Editing Factual Associations in GPT. *NeurIPS*, 2022.

Merullo, J., Eickhoff, C., and Pavlick, E. Circuit Component Reuse Across Tasks in Transformer Language Models. In *ICLR*, 2021.

Michel, P., Levy, O., and Neubig, G. Are Sixteen Heads Really Better Than One? In *NeurIPS*, 2019.

Mihaylov, T., Clark, P., Khot, T., and Sabharwal, A. Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering. In *EMNLP*, 2018.

Molchanov, P., Tyree, S., Karras, T., Aila, T., and Kautz, J. Pruning Convolutional Neural Networks for Resource Efficient Inference. In *ICLR*, 2017.

OpenAI. Introducing ChatGPT. https://openai.com/blog/chatgpt, 2022.

OpenAI. GPT-4 Technical Report, 2023.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. Training Language Models to Follow Instructions With Human Feedback. In *NeurIPS*, 2022.

Panigrahi, A., Saunshi, N., Zhao, H., and Arora, S. Task-Specific Skill Localization in Fine-tuned Language Models. In *ICML*, 2023.

Qi, X., Huang, K., Panda, A., Wang, M., and Mittal, P. Visual Adversarial Examples Jailbreak Aligned Large Language Models. In *AAAI*, 2024a.

Qi, X., Zeng, Y., Xie, T., Chen, P.-Y., Jia, R., Mittal, P., and Henderson, P. Fine-tuning Aligned Language Models Compromises Safety, Even When Users Do Not Intend To! In *ICLR*, 2024b.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language Models are Unsupervised Multitask Learners. *OpenAI blog*, 2019.

Rafailov, R., Sharma, A., Mitchell, E., Manning, C. D., Ermon, S., and Finn, C. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS*, 2023.

Ribeiro, M. T., Singh, S., and Guestrin, C. "Why Should I Trust You?" Explaining the Predictions of Any Classifier. In *KDD*, 2016.

Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y. WinoGrande: An Adversarial Winograd Schema Challenge at Scale . *Communications of the ACM*, 2021.

Sanh, V., Wolf, T., and Rush, A. Movement Pruning: Adaptive Sparsity by Fine-Tuning. *NeurIPS*, 2020.

Schotthöfer, S., Zangrando, E., Kusch, J., Ceruti, G., and Tudisco, F. Low-Rank Lottery Tickets: Finding Efficient Low-Rank Neural Networks via Matrix Differential Equations. *NeurIPS*, 2022.

Sharma, P., Ash, J. T., and Misra, D. The Truth is in There: Improving Reasoning in Language Models with Layer-Selective Rank Reduction. *arXiv preprint arXiv:2312.13558*, 2023.

Shen, X., Chen, Z., Backes, M., Shen, Y., and Zhang, Y. "Do Anything Now": Characterizing and Evaluating In-The-Wild Jailbreak Prompts on Large Language Models. *arXiv preprint arXiv:2308.03825*, 2023.

Shrikumar, A., Greenside, P., and Kundaje, A. Learning Important Features Through Propagating Activation Differences. In *ICML*, 2017.

Springenberg, J., Dosovitskiy, A., Brox, T., and Riedmiller, M. Striving for Simplicity: The All Convolutional Net. In *ICLR (workshop track)*, 2015.

Sun, M., Liu, Z., Bair, A., and Kolter, J. Z. A Simple and Effective Pruning Approach for Large Language Models. In *ICLR*, 2024.

Sun, Z., Shen, Y., Zhou, Q., Zhang, H., Chen, Z., Cox, D., Yang, Y., and Gan, C. Principle-Driven Self-Alignment of Language Models from Scratch with Minimal Human Supervision. In *NeurIPS*, 2023.

Sundararajan, M., Taly, A., and Yan, Q. Axiomatic Attribution for Deep Networks. In *ICML*, 2017.

Taori, R., Gulrajani, I., Zhang, T., Dubois, Y., Li, X., Guestrin, C., Liang, P., and Hashimoto, T. B. Stanford Alpaca: An Instruction-following LLaMA model. https://github.com/tatsu-lab/stanford_alpaca, 2023.

Team, G., Anil, R., Borgeaud, S., Wu, Y., Alayrac, J.-B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A. M., Hauth, A., et al. Gemini: A Family of Highly Capable Multimodal Models. *arXiv preprint arXiv:2312.11805*, 2023.

Tjoa, E. and Guan, C. A Survey on Explainable Artificial Intelligence (XAI): Towards Medical XAI. *IEEE transactions on neural networks and learning systems*, 2020.

Todd, E., Li, M. L., Sharma, A. S., Mueller, A., Wallace, B. C., and Bau, D. Function Vectors in Large Language Models. *arXiv preprint arXiv:2310.15213*, 2023.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. LLaMA: Open and Efficient Foundation Language Models. *arXiv preprint arXiv:2302.13971*, 2023a.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. Llama 2: Open Foundation and Fine-Tuned Chat Models. *arXiv preprint arXiv:2307.09288*, 2023b.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *ICLR*, 2019.

Wang, X., Wen, K., Zhang, Z., Hou, L., Liu, Z., and Li, J. Finding Skill Neurons in Pre-trained Transformer-based Language Models. In *EMNLP*, 2022.

Wang, Z., Wohlwend, J., and Lei, T. Structured Pruning of Large Language Models. In *EMNLP*, 2020.

Wei, A., Haghtalab, N., and Steinhardt, J. Jailbroken: How Does LLM Safety Training Fail? In *NeurIPS*, 2023.

Wei, J., Bosma, M., Zhao, V. Y., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., and Le, Q. V. Finetuned Language Models Are Zero-Shot Learners. In *ICLR*, 2022.

Wen, W., Wu, C., Wang, Y., Chen, Y., and Li, H. Learning Structured Sparsity in Deep Neural Networks. *NeurIPS*, 29, 2016.

Xia, M., Zhong, Z., and Chen, D. Structured Pruning Learns Compact and Accurate Models. In *ACL*, 2022.

Xia, M., Gao, T., Zeng, Z., and Chen, D. Sheared LLaMA: Accelerating Language Model Pre-training via Structured Pruning. In *ICLR*, 2024.

Xu, J., Lee, A., Sukhbaatar, S., and Weston, J. Some things are more CRINGE than others: Preference Optimization with the Pairwise Cringe Loss. *arXiv preprint arXiv:2312.16682*, 2023.

Yang, K., Klein, D., Celikyilmaz, A., Peng, N., and Tian, Y. RLCD: Reinforcement Learning from Contrastive Distillation for LM Alignment. In *ICLR*, 2024.

Yang, X., Wang, X., Zhang, Q., Petzold, L., Wang, W. Y., Zhao, X., and Lin, D. Shadow Alignment: The Ease of Subverting Safely-Aligned Language Models. *arXiv preprint arXiv:2310.02949*, 2023.

Yong, Z. X., Menghini, C., and Bach, S. Low-Resource Languages Jailbreak GPT-4. In *Socially Responsible Language Modelling Research*, 2023.

Yuan, W., Pang, R. Y., Cho, K., Sukhbaatar, S., Xu, J., and Weston, J. Self-Rewarding Language Models. *arXiv preprint arXiv:2401.10020*, 2024a.

Yuan, Y., Jiao, W., Wang, W., Huang, J.-t., He, P., Shi, S., and Tu, Z. GPT-4 Is Too Smart To Be Safe: Stealthy Chat with LLMs via Cipher. In *ICLR*, 2024b.

Yuan, Z., Shang, Y., Song, Y., Wu, Q., Yan, Y., and Sun, G. ASVD: Activation-aware Singular Value Decomposition for Compressing Large Language Models. *arXiv preprint arXiv:2312.05821*, 2023.

Zeiler, M. D. and Fergus, R. Visualizing and Understanding Convolutional Networks. In *ECCV*, 2014.

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. HellaSwag: Can a Machine Really Finish Your Sentence? In *ACL*, 2019.

Zeng, Y., Lin, H., Zhang, J., Yang, D., Jia, R., and Shi, W. How Johnny Can Persuade LLMs to Jailbreak Them: Rethinking Persuasion to Challenge AI Safety by Humanizing LLMs. *arXiv preprint arXiv:2401.06373*, 2024.

Zhan, Q., Fang, R., Bindu, R., Gupta, A., Hashimoto, T., and Kang, D. Removing RLHF Protections in GPT-4 via Fine-Tuning. *arXiv preprint arXiv:2311.05553*, 2023.

Zhang, M., Shen, C., Yang, Z., Ou, L., Yu, X., Zhuang, B., et al. Pruning Meets Low-Rank Parameter-Efficient Fine-Tuning. *arXiv preprint arXiv:2305.18403*, 2023a.

Zhang, Q., Chen, M., Bukharin, A., He, P., Cheng, Y., Chen, W., and Zhao, T. AdaLoRA: Adaptive Budget Allocation for Parameter-Efficient Fine-Tuning. In *ICLR*, 2023b.

Zhao, H., Chen, H., Yang, F., Liu, N., Deng, H., Cai, H., Wang, S., Yin, D., and Du, M. Explainability for Large Language Models: A Survey. *ACM Transactions on Intelligent Systems and Technology*, 2023.

Zhao, M., Lin, T., Mi, F., Jaggi, M., and Schütze, H. Masking as an Efficient Alternative to Finetuning for Pretrained Language Models. In *EMNLP*, 2020.

Zhu, S., Zhang, R., An, B., Wu, G., Barrow, J., Wang, Z., Huang, F., Nenkova, A., and Sun, T. AutoDAN: Automatic and Interpretable Adversarial Attacks on Large Language Models. *arXiv preprint arXiv:2310.15140*, 2023.

Ziegler, D. M., Stiennon, N., Wu, J., Brown, T. B., Radford, A., Amodei, D., Christiano, P., and Irving, G. Fine-Tuning Language Models From Human Preferences. *arXiv preprint arXiv:1909.08593*, 2019.

Zou, A., Phan, L., Chen, S., Campbell, J., Guo, P., Ren, R., Pan, A., Yin, X., Mazeika, M., Dombrowski, A.-K., et al. Representation Engineering: A Top-Down Approach to AI Transparency. *arXiv preprint arXiv:2310.01405*, 2023a.

Zou, A., Wang, Z., Kolter, J. Z., and Fredrikson, M. Universal and Transferable Adversarial Attacks on Aligned Language Models. *arXiv preprint arXiv:2307.15043*, 2023b.

A. Related Work

A.1. Alignment and Jailbreak

Alignment refers to the process of ensuring a machine learning (ML) model's behavior conforms to human values. For example, pretrained language models are usually not aligned with human objectives – they cannot follow users' instructions, and could potentially generate harmful and incorrect content. During the alignment stage, practitioners would employ Instruction Tuning (Wei et al., 2022; Ouyang et al., 2022; Touvron et al., 2023b), and Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022; Touvron et al., 2023b; Bai et al., 2022a) to enforce the language models to be *helpful, harmless, and honest* (the HHH principle) (Askell et al., 2021). Aligned LLMs (e.g., OpenAI ChatGPT (OpenAI, 2023) and Anthropic Claude (Anthropic, 2023a;b)), as a result, will follow human values and refuse to respond to harmful requests. Recent work (Rafailov et al., 2023; Xu et al., 2023; Dai et al., 2024; Yang et al., 2024; Li et al., 2024; Yuan et al., 2024a) propose more effective and efficient alignment alternatives to RLHF. As some examples, Direct Preference Optimization (DPO) (Rafailov et al., 2023) directly fine-tunes language models on human preference data, eliminating the need to train a reward model and conduct reinforcement learning; Self-Rewarding (Yuan et al., 2024a) uses the language model itself as a reward model to curate labeled preference data, and then align the language model with DPO in an iterative way; (Dai et al., 2024) proposes to decouple the goal of safety and helpfulness during alignment, similar to the decoupling goal of our work.

While harmful instructions that are plain and direct would be rejected by aligned LLMs, researchers and communities have identified ways to *bypass or remove* the safety guardrails enforced by LLM alignment efforts — namely "jailbreaking" LLMs. More specifically, *jailbreaking* is a series of attacks where an adversary would coax or enforce the model to deviate from its ethical guidelines. In practical, jailbreak attackers would either employ *adversarial prompts* (Liu et al., 2023; Zou et al., 2023b; Yuan et al., 2024b; Liu et al., 2024; Shen et al., 2023; Yong et al., 2023; Mehrotra et al., 2023) or manipulate model *decoding* process (Huang et al., 2024) to bypass LLM safety alignment. Moreover, when having *fine-tuning* access of LLMs, adversaries (Qi et al., 2024b; Yang et al., 2023; Zhan et al., 2023) could directly remove the guardrails. A jailbroken LLM would provide harmful responses to comply with users' harmful requests, which they otherwise would simply reject (due to the ethical guidelines injected by alignment) — this could subsequently pose serious safety risks in the real world, since LLMs can directly deliver various harmfulness to individuals and the society.

A.2. Identifying Task-Specific Regions in Large Language Models

Attributing the model's behavior to the model's weights is a classic research question in explainable machine learning (Tjoa & Guan, 2020; Burkart & Huber, 2021; Ali et al., 2023). Previous studies in pre-transformer eras have explored various approaches to identify task-specific neurons within models, to better interpret and control the model's behavior. Popular techniques mainly include perturbation-based methods that perturb the input of a model and observe the changes in the output of the model (Zeiler & Fergus, 2014; Ribeiro et al., 2016), and gradient-based methods that compute an importance score for model weights based on the results from task-specific back propagation (Springenberg et al., 2015; Bach et al., 2015; Sundararajan et al., 2017; Shrikumar et al., 2017; Lundberg & Lee, 2017). However, it was only recently that these methods have been rigorously applied to modern transformer models (Madsen et al., 2022; Zhao et al., 2023; Maini et al., 2023).

Probing has emerged as a method for understanding the knowledge encoded in transformers, particularly large language models (Adi et al., 2016; Conneau et al., 2018; Hewitt & Liang, 2019). To perform probing, the model representations and model parameters are fed into a probe classifier (Belinkov, 2022), whose task is to identify certain linguistic properties or reasoning abilities acquired by the model. For instance, previous work has adopted probing-based method to localize truthfulness (Li et al., 2023a; Campbell et al., 2023), factuality (Meng et al., 2022; Geva et al., 2023), toxicity (Lee et al., 2024), and knowledge (Burns et al., 2023; Todd et al., 2023) in LLMs. More recently, Zou et al. (2023a) propose a similar approach to probing: instead of training a classifier, they employ an unsupervised approach, specifically singular value decomposition, to identify significant directions in the representation space. They then demonstrate that these directions can predict and influence the behavior of LLMs.

In addition to the importance-score-based and probing-based methods discussed above, recent studies have also investigated a range of techniques to pinpoint task-specific neurons in transformers. These techniques address various aspects of the model, including linguistic properties (Dalvi et al., 2019; Antverg & Belinkov, 2021), general capabilities (Lan & Barez, 2023; Gurnee et al., 2024; Merullo et al., 2021), fine-tuning (Panigrahi et al., 2023; Lubana et al., 2023), and prompt tuning (Wang et al., 2022).

The closest concurrent work to ours are Lee et al. (2024) and Jain et al. (2023). Lee et al. (2024) investigate the representation and elicitation of toxicity in a GPT-2 (Radford et al., 2019) model, and explores via probing how aligning the model using Direct Preference Optimization (DPO) (Rafailov et al., 2023) mitigates toxicity. Their findings suggest that DPO does not eliminate the model's ability to generate toxic outputs, but rather learns to bypass the regions that elicit toxicity. While Lee et al. (2024) reveal the fragility of model alignment by probing the GPT-2 model at the granularity of per attention head level, our study examines the more advanced Llama family models (Touvron et al., 2023b) using per-neuron or per-rank attribution, which is more relevant to real-world applications and allows for a more fine-grained analysis. Jain et al. (2023) investigate the impact of fine-tuning on LLMs, using methods such as probing and data-agnostic structured pruning. They suggest that fine-tuning model weights might create a 'safety wrapper' around core models, rendering the effects of safety alignment easily reversible. In contrast to their approach which operates on the transformer block level, our study examines the models at a more fine-grained neuron level and rank level.

A.3. Low Rank Compression

Our work borrows insight from Hsu et al. (2021); Schotthöfer et al. (2022); Zhang et al. (2023b); Yuan et al. (2023); Li et al. (2023b). We address two similar methods (Yuan et al., 2023; Hsu et al., 2021) and point out their differences from ActSVD. Yuan et al. (2023) propose ASVD (Activation-aware Singular Value Decomposition) as a low-rank compression technique for language models. Their method performs SVD on WS, where S is a diagonal matrix with S_{ii} given by the norm of the activations

$$S_{ii} = \left(\frac{1}{n}\sum_j |X_{ij}|\right)^{\alpha} \quad \text{or} \quad \left(\max_j |X_{ij}|\right)^{\alpha}.$$

Hsu et al. (2021) propose FWSVD (Fisher-Weighted SVD) as a low-rank compression technique, where the SVD is applied to $\hat{I}W$. Here

$$\hat{I} = \text{diag}(\sqrt{\hat{I}_{W_1}}, \ldots, \sqrt{\hat{I}_{W_d}}), \quad \hat{I}_{W_i} = \sum_j \hat{I}_{W_{ij}}.$$

The $\hat{I}_{W_{ij}}$ is defined as the Fisher information of the loss function with respect to the weight entry W_{ij}

$$\hat{I}_{W_{ij}} = \frac{1}{D}\sum_i \left(\partial_{W_{ij}} \mathcal{L}(x_i; W_{ij})\right)^2.$$

In contrast, for the proposed method ActSVD, we perform SVD on WX_{in}, where X_{in} is the stack of all activations, i.e., $X_{\text{in}} = [X_1, \ldots, X_n] \in \mathbb{R}^{d_{\text{in}} \times n}$.

A.4. Pruning

Our approach to attribution aligns closely with techniques used in neural network pruning. Our SNIP method (Lee et al., 2019) resembles more closely with unstructured pruning techniques, which are designed to establish criteria based on weight magnitude, activations, or network gradients for removing individual weights from a network (Han et al., 2016; Molchanov et al., 2017; Frankle & Carbin, 2018; Chen et al., 2020; Sanh et al., 2020; Zhao et al., 2020; Cao et al., 2021; Guo et al., 2021). These unstructured pruning methods have been adapted for use in large language models, as seen in Wanda (Sun et al., 2024), SparseGPT (Frantar & Alistarh, 2023).

Broadly speaking, the low-rank compression techniques are akin to structured pruning approaches, with a focus on identifying important structured subnetworks. In computer vision settings, it is common to remove channels or filters (Li et al., 2017; Molchanov et al., 2017; Wen et al., 2016; He et al., 2017; Luo et al., 2017; Liu et al., 2017) from convolutional neural networks. Structured pruning of language models involves removing heads, dimensions, or ranks (Michel et al., 2019; Wang et al., 2020; Lagunas et al., 2021; Xia et al., 2022; Ma et al., 2023; Zhang et al., 2023b;a; Chen et al., 2023; Xia et al., 2024; Ashkboos et al., 2024).

While pruning is commonly employed for model compression to decrease model sizes, our work adopts similar techniques to identify critical regions responsible for safety.

B. Experimental Details

Compute configurations All the experiments are done with four AMD EPYC 7J13 64-core CPUs and a single NVIDIA A100-80G GPU. During the experiments, we utilize vLLM (Kwon et al., 2023) for faster decoding. The typical GPU hours for different experiments are listed in Table 4.

Model Name	Attribution unit	Experiment type	GPU hours
`Llama2-7B-chat`	Neuron	Pruning Top/Least Safety Neurons	0.2
		Pruning with Set Difference	0.2
	Rank	Removing Top/Least Safety Ranks	0.5
		Removing with Orthogonal Projection	1.0
`Llama2-13B-chat`	Neuron	Pruning Top/Least Safety Neurons	0.5
		Pruning with Set Difference	0.5
	Rank	Removing Top/Least Safety Ranks	0.8
		Removing with Orthogonal Projection	2.0

Table 4. Typical GPU hours of different experiment types.

Details for pruning For all the methods in the paper, we adopt block-wise pruning as Sun et al. (2024), where we start from the first Transformer block in Llama. After pruning the 7 linear layers in the current block (self_attn.q, self_attn.k, self_attn.v, self_attn.o, mlp.up, mlp.gate, mlp.down), we *recompute* the outputs of the current block and continue to the next block.

For the neuron-level attribution, we use output-wise pruning following Sun et al. (2024), as the authors observed that pruning per output has better performance for language models. Specifically, after we obtain the score matrix $I(W)$, for a specific sparsity ratio $p\%$, we set $p\%$ of the weights to zero *independently for each row* of the matrix W.

Collection of safety and utility dataset Table 5 provides more details for the safety and utility datasets we use in our experiments. During the experiment, we sample 128 (prompt, response) pairs in computing the importance score or projection matrix.

Dataset Name	Data sources	Number of (prompt, response) pairs	Average Sequence Length
Safety-full	$\texttt{AdvBench}_{\text{attr}}$ (prompt)	$7,220$	175.68
Safety-short	$\texttt{AdvBench}_{\text{attr}}$ (prompt)	$7,220$	48.78
Utility	`Alpaca-Cleaned`	$45,874$	118.50

Table 5. The basic information of datasets used for attribution in our experiments.

Repeat times To mitigate the potential variability introduced by random seeds, we repeat our experiments on Section 4.2 three times with different random seeds. In our figure, we plot the mean value μ for each data point. To represent variability, we shade the area between $[\mu - \sigma, \mu + \sigma]$, where σ denotes the standard deviation corresponding to each point.

The probing baseline We adopt a similar probing setup used in Li et al. (2023a) for identifying safety-critical neurons in this work. Specifically, we feed the model with all 420 harmful instructions from $\text{AdvBench}_{\text{attr}}$, as well as 420 harmless instructions randomly sampled from the utility dataset. We collect the activation outputs of every internal attention head for these instructions. This collected data is then split into two sets, with a $5:2$ ratio for the training split and the validation split, respectively. For each attention head, we train a linear classifier on the training split using its activation inputs to distinguish between activations resulting from harmful and harmless instructions. We then evaluate the accuracy of the classifier on the validation split, which indicates the relevance of the attention head in distinguishing between harmful and harmless instructions.

The adversarial suffixes We run the GCG attack (Zou et al., 2023b) for 500 iterations, with adversarial sting initiated as "! !". For optimization, we use a batch size of 512, top-k as 256, with a joint optimization over Llama2 family (Touvron et al., 2023b) and Vicuna (Chiang et al., 2023) models[10], with their system prompts removed, for three independent trails. We then identify the top three suffixes with the highest attack success rates on `AdvBench`, and use them in our evaluation. For ethical reasons, we refrain from disclosing these suffixes to prevent potential misuse.

The adversarial decoding For evaluating $\text{ASR}_{\text{Adv-Decoding}}$, we configure the sampling temperature to 1.0 when generating responses from $\texttt{AdvBench}_{\text{eval}}$. For each harmful prompt in `AdvBench`, we perform sampling 5 times. An attack is considered successful if at least one of the sampled responses is deemed harmful.

The details of the zero-shot tasks in evaluating utility

1. ARC-Challenge:
   (a) Downstream Task: Science Question Answering.
   (b) Description: The ARC-Challenge metric evaluates the performance of models on the ARC-Challenge subset of the AI2 Reasoning Challenge dataset, which consists of grade-school science questions that require complex reasoning and understanding of scientific concepts [11].
2. HellaSWAG:
   (a) Downstream Task: Commonsense Reasoning
   (b) Description: HellaSWAG is a dataset for evaluating commonsense reasoning in AI systems. It consists of context and multiple-choice endings, where the task is to predict the most plausible ending. The dataset is designed to test a model's ability to reason about everyday scenarios[12].
3. OpenBookQA:
   (a) Downstream Task: Open-Book Question Answering
   (b) Description: OpenBookQA aims to promote research in advanced question-answering, probing a deeper understanding of both the topic (with salient facts summarized as an open book, also provided with the dataset) and the language it is expressed in. In particular, it contains questions that require multi-step reasoning, use of additional common and commonsense knowledge, and rich text comprehension. OpenBookQA is a new kind of question-answering dataset modeled after open book exams for assessing human understanding of a subject[13].
4. WiNoGrande:
   (a) Downstream Task: Commonsense Reasoning Description: WiNoGrande is a dataset for evaluating large-scale commonsense reasoning. It is inspired by Winograd Schema Challenge (Levesque, Davis, and Morgenstern 2011), but adjusted to improve the scale and robustness against the dataset-specific bias. Formulated as a fill-in-a-blank task with binary options, the goal is to choose the right option for a given sentence which requires commonsense reasoning[14].
5. BoolQ:
   (a) Downstream Task: Yes/No Question Answering
   (b) Description: BoolQ is a question answering dataset for yes/no questions containing 15942 examples. These questions are naturally occurring —they are generated in unprompted and unconstrained settings. Each example is a triplet of (question, passage, answer), with the title of the page as optional additional context. The text-pair classification setup is similar to existing natural language inference tasks[15].
6. RTE (Recognizing Textual Entailment):

[10] According to Zou et al. (2023b), incorporating a broader range of models during training enhances the effectiveness of attacks.
[11] More details are available at `https://allenai.org/data/arc`.
[12] More details are available at `https://huggingface.co/datasets/Rowan/hellaswag`.
[13] More details are available at `https://huggingface.co/datasets/allenai/openbookqa`.
[14] More details are available at `https://huggingface.co/datasets/winogrande`.
[15] More details are available at `https://github.com/google-research-datasets/boolean-questions`.

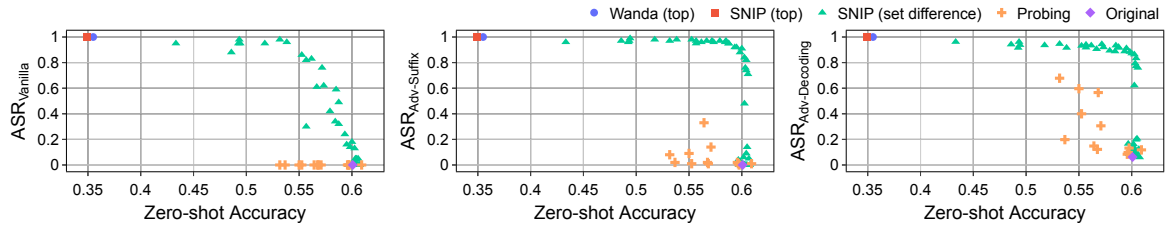
(a) Downstream Task: Textual Entailment

(b) Description: RTE is a task that involves determining whether a given hypothesis can logically be inferred from a given premise. The dataset consists of pairs of sentences, and the task is to classify each pair as either "entailment" (the hypothesis follows from the premise) or "not entailment" (the hypothesis does not follow from the premise) [16].

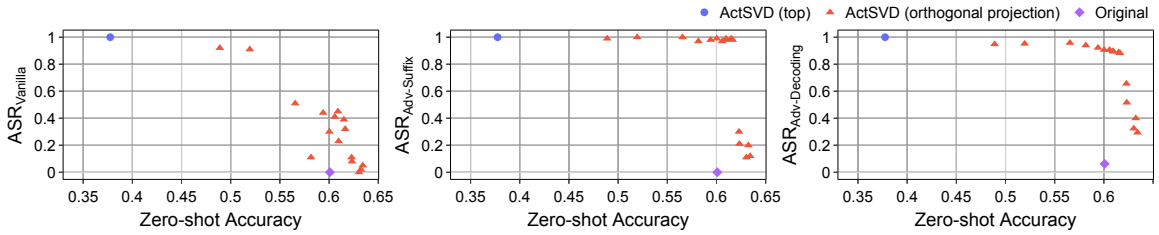
C. More Experimental Results

C.1. More results in `Llama2-13B-chat`

We plot the results of removing the most safety-critical neurons and ranks on Figure 5 and the results of removing the least safety-critical neurons and ranks on Figure 6.



(a) Removing Safety-Critical **Neurons**



(b) Removing Safety-Critical **Ranks**

Figure 5. ASR and accuracy after removing safety-critical neurons in `Llama2-13B-chat` identified by **(a)** different methods in Section 3.2 with **sparsity constraint** $< 3\%$; **(b)** different methods in Section 3.3 with **ranks of the weight updates** $\leq$ **120** (out of 5120). Among all methods, disentangling safety from utility (set difference for neurons and orthogonal projection for ranks) mostly effectively identify the safety-critical regions, with safety severely compromised while utility retains.

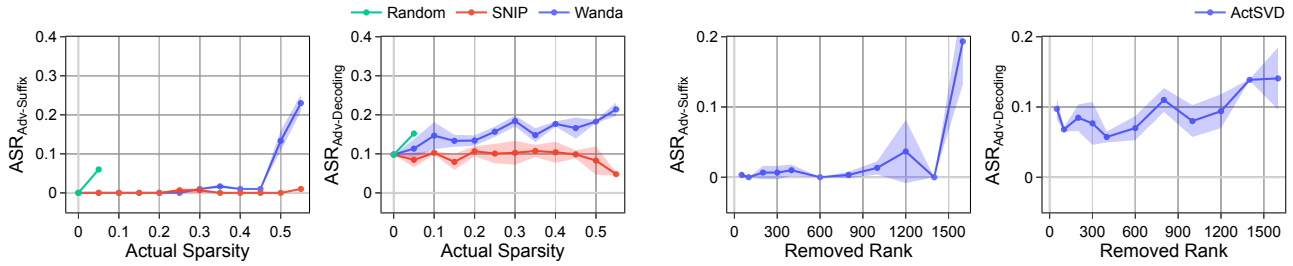
In accordance with the results on `Llama2-7B-chat` (see Section 4), from Figure 5, we observe similar results on `Llama2-13B-chat`:

- Removing safety-critical neurons using set difference, or removing safety-critical ranks using orthogonal projection, is effective in destroying the model's safety while preserving utility.
- Removing top safety neurons or ranks severely hurts utility.
- Set difference with SNIP score consistently outperforms the attention head probing baseline.

However, from Figure 6, we observe different curves from the results for `Llama2-7B-chat`.

- The exclusion of neurons using SNIP and ActSVD deemed least critical for safety slightly enhances robustness against adversarial decoding attacks, i.e., when actual sparsity > 0.4 & removed rank < 300.
- In contrast, removing neurons according to the least Wanda scores hurts the adversarial robustness.
- Different from `Llama2-7B-chat`, we see that the original `Llama2-13B-chat` has zero $\text{ASR}_{\text{Adv-Suffix}}$. Removing less than 45% of neurons or less than 750 ranks that are least critical for safety *maintains* the robustness against adversarial suffixes at a nearly 0 ASR. One potential reason behinds the phenomenon is that the adversarial suffixes are obtained using 7B models and they cannot transfer to `Llama2-13B-chat`. It may be possible that the trend between the `Llama2-13B-chat` and `Llama2-7B-chat` models becomes more aligned with optimized suffixes.

[16] More details are available at https://huggingface.co/datasets/nyu-mll/glue#rte.



(a) Removing the least Safety-Critical **Neurons**

(b) Removing the least Safety-Critical **Ranks**

Figure 6. Impact on ASR under adversaries in `Llama2-13B-chat` when: (a) Removing neurons with *the lowest* safety importance scores and (b) Removing *the least* safety-critical ranks identified by ActSVD. Both the importance score and safety-critical ranks are calculated based on safety short. All plotted points have reasonable utility with accuracy > 0.5. The exclusion of neurons (using SNIP) and ranks deemed least critical for safety slightly enhances robustness against adversarial decoding attacks (when actual sparsity < 0.4 & removed rank < 300), and it maintains the robustness at a nearly 0 ASR against adversarial suffixes.

C.2. Ablation Study between safety-full dataset and safety-short dataset

Performance of set difference As shown in Figure 7, the trends in ASR versus accuracy for both the safety-full and safety-short attribution datasets are similar. This observation implies that utilizing judgment-only data is as effective as using the full response for identifying safety-critical neurons.

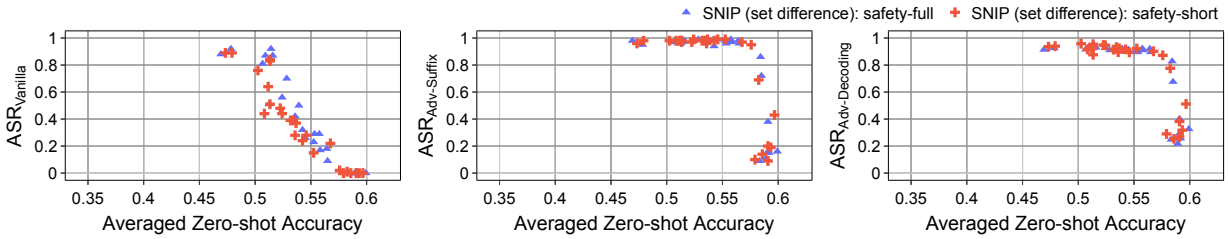


Figure 7. The relationship between ASR and Averaged Zero-shot Accuracy with set difference pruning methods on `Llama2-7B-chat`, using safety and safety-short datasets. The model's performance is very similar in safety and safety-short.

Performance of pruning the least safety-critical region As shown in Figure 8, when pruning the least safety-critical region, compared to safety-full dataset, using safety-short dataset exhibits a more significant change in both $\text{ASR}_{\text{Adv-Suffix}}$ and $\text{ASR}_{\text{Adv-Decoding}}$. We also observe that $\text{ASR}_{\text{Vanilla}}$ remains close to zero for actual sparsity levels between 0 and 0.55. Therefore, we only report results for $\text{ASR}_{\text{Adv-Suffix}}$ and $\text{ASR}_{\text{Adv-Decoding}}$, with safety-short in Section 4.2.

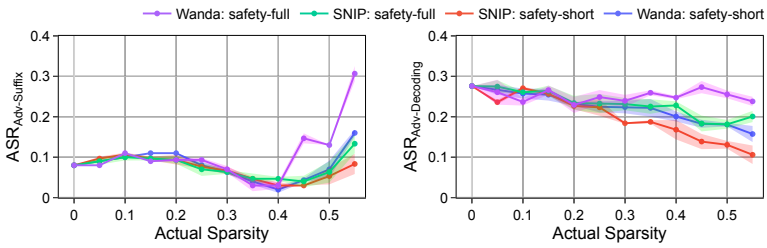


Figure 8. Comparison between using safety-full and safety-short when pruning the least safety-critical neurons on `Llama2-7B-chat`. Compared to safety-full dataset, using safety-short dataset exhibits a more significant change.

C.3. More Results for Disentanglement Methods

More results for (p, q) combinations in set difference. We conduct a comprehensive study exploring the search space for (p, q) values ranging between 0.1 and 90. Complementing Figure 2a and Figure 5a, Table 6 presents the top (p, q) combinations along with the utility and safety measures for the resulting models on `Llama2-7B-chat` and `Llama2-13B-chat`. In scenarios where the actual sparsity is less than 1%, the model maintains a low $\text{ASR}_{\text{Vanilla}}$, typically under 0.3. However, its $\text{ASR}_{\text{Adv-Suffix}}$ and $\text{ASR}_{\text{Adv-Decoding}}$ nearly reach to 1. In contrast, when the actual sparsity

$\in (1\%, 3\%)$., the model approaches a value close to 1 for all three ASR variants. Notably, across all cases outlined in Table 6, the model consistently maintains utility, with an average accuracy greater than 0.5. These findings indicate that the optimal range for the (p, q) parameters lies between 3 and 9, especially when the values of p and q are similar.

p	q	Actual Sparsity	$\text{ASR}_{\text{Vanilla}}$	$\text{ASR}_{\text{Adv-Suffix}}$	$\text{ASR}_{\text{Adv-Decoding}}$	Averaged Accuracy
Actual Sparsity $< 1\%$						
1	1	0.63%	0.10	0.97	0.86	0.54
2	1	0.46%	0.06	0.94	0.84	0.55
4	2	0.78%	0.09	0.97	0.88	0.56
7	3	0.86%	0.06	0.97	0.89	0.58
3	2	0.94%	0.21	0.97	0.89	0.55
Actual Sparsity $\in (1\%, 3\%)$						
4	4	2.03%	0.81	0.97	0.92	0.51
5	5	2.41%	0.92	0.97	0.95	0.51
6	5	2.10%	0.70	0.99	0.91	0.53
6	6	2.75%	0.87	0.97	0.93	0.51
9	8	2.99%	0.87	0.98	0.93	0.52

(a) `Llama2-7B-chat`

p	q	Actual Sparsity	$\text{ASR}_{\text{Vanilla}}$	$\text{ASR}_{\text{Adv-Suffix}}$	$\text{ASR}_{\text{Adv-Decoding}}$	Averaged Accuracy
Actual Sparsity $< 1\%$						
1	1	0.66%	0.30	0.82	0.83	0.60
5	2	0.72%	0.18	0.91	0.87	0.60
7	2	0.53%	0.13	0.84	0.86	0.60
7	3	0.93%	0.16	0.92	0.91	0.60
8	3	0.82%	0.14	0.88	0.88	0.60
Actual Sparsity $\in (1\%, 3\%)$						
3	3	1.72%	0.96	0.98	0.92	0.54
4	3	1.46%	0.86	0.98	0.93	0.55
4	4	2.15%	0.98	0.97	0.95	0.53
6	6	2.91%	0.95	0.98	0.94	0.52
9	7	2.59%	0.83	0.96	0.93	0.56

(b) `Llama2-13B-chat`

Table 6. Performance of `Llama2-7B-chat` (a) and `Llama2-13B-chat` (b) with safety-critical neurons removed, identified through set difference between top-$q\%$ safety and top-$p\%$ utility neurons, across various (p, q) combinations. The ideal range for (p, q) is $[3, 9]$, especially when p and q are closely matched.

More results for (r^u, r^s) combinations in orthogonal projection. Note that the ranks of the weight matrices of the linear layers are $R = 4096$ for `Llama2-7B-chat` and $R = 5120$ for `Llama2-13B-chat`. We perform a grid search for the parameters r^u and r^s, spanning a range from 50 to 4000 for `Llama2-7B-chat` and from 1200 to 5000 for `Llama2-13B-chat`. As an extension to Figure 2b and Figure 5b, Table 7 presents the top five combinations of r^u and r^s along with the utility and safety metrics for the models tested on `Llama2-7B-chat` and `Llama2-13B-chat` model. The results indicate that setting r^s close to R, especially when r^u closes to r^s, proves to be particularly effective.

C.4. Probing Accuracy Distributions

We also analyze the accuracy of linear probers trained on all 1024 attention heads from `Llama2-7B-chat` (Figure 9a) and 1600 attention heads from `Llama2-13B-chat` (Figure 9b). The results show that around half of the attention heads achieve very high probing accuracy (i.e., > 0.95) in distinguishing between harmful and harmless instructions. Notably, even the attention heads with the lowest probing accuracy show significant effectiveness – 0.78 for `Llama2-7B-chat` and 0.74 for `Llama2-13B-chat`. Additionally, transformer blocks located in the middle typically demonstrate higher probing accuracy compared to those at the beginning or end.

This pattern of high accuracy suggests that making safety judgments at the level of individual attention heads is relatively

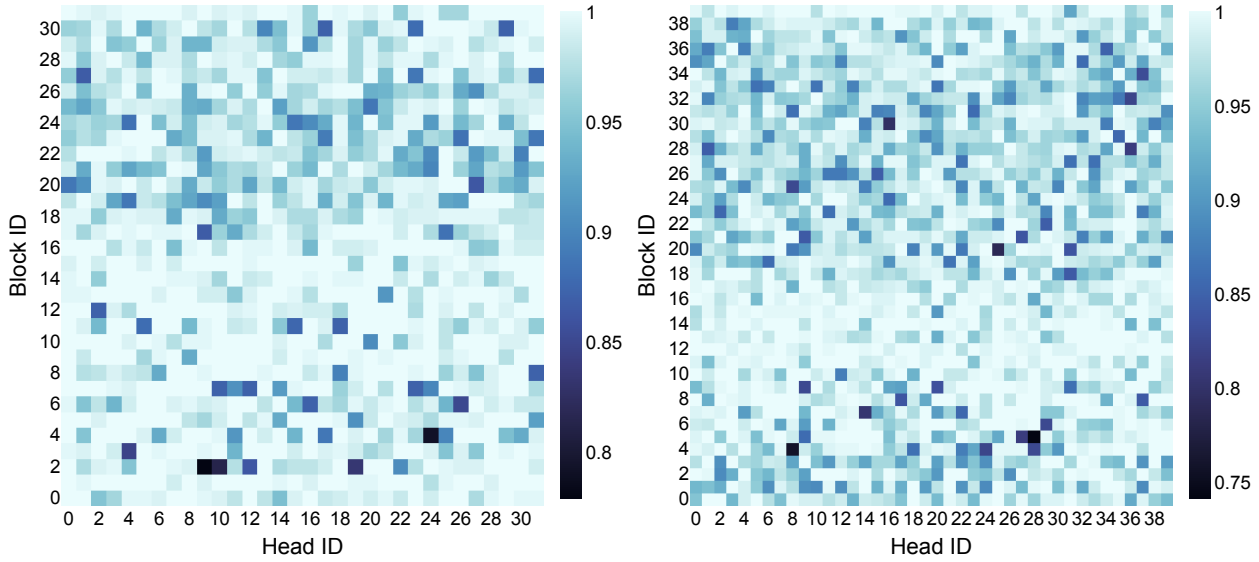
r^u	r^s	$\min(r^u, R - r^s)$	**ASR$_{\text{Vanilla}}$**	**ASR$_{\text{Adv-Suffix}}$**	**ASR$_{\text{Adv-Decoding}}$**	**Averaged Accuracy**
3450	4000	96	0.67	1.00	0.88	0.59
3550	4000	96	0.68	0.99	0.90	0.59
3950	4090	6	0.71	0.97	0.91	0.59
4000	4090	6	0.71	0.97	0.92	0.58
4080	4090	6	0.65	0.98	0.94	0.57

(a) `Llama2-7B-chat` ($R = 4096$)

r^u	r^s	$\min(r^u, R - r^s)$	**ASR$_{\text{Vanilla}}$**	**ASR$_{\text{Adv-Suffix}}$**	**ASR$_{\text{Adv-Decoding}}$**	**Averaged Accuracy**
3450	5000	120	0.32	0.98	0.88	0.62
3600	5000	120	0.41	0.97	0.91	0.61
3900	5000	120	0.30	0.99	0.91	0.60
3750	5000	120	0.39	0.99	0.89	0.62
4400	5000	120	0.91	1.00	0.95	0.52

(b) `Llama2-13B-chat` ($R = 5120$)

Table 7. Performance of `Llama2-7B-chat` (a) and `Llama2-13B-chat` (b) model with safety-critical ranks removed by doing orthogonal projection between utility projection matrix Π^u and safety projection matrix Π^s, across various (r^u, r^s) combinations. Setting r^s close to R, especially when r^u closes to r^s, proves to be particularly effective.



(a) `Llama2-7B-chat` (b) `Llama2-13B-chat`

Figure 9. The probing accuracy distribution across different attention heads inside `Llama2-7B-chat` and `Llama2-13B-chat`. Around half of the attention heads achieve very high probing accuracy (i.e., > 0.95) in distinguishing between harmful and harmless instructions.

straightforward due to their effective representational capacity. Therefore, our study's focus on finer granularities, such as neurons or ranks, is essential for the precise localization of safety-critical regions.

D. Proof of the Optimality of ActSVD

Lemma 1. *Let $X_{\text{in}} \in \mathbb{R}^{d_{\text{in}} \times n}$. Let $\widehat{W}$ be the solution to the following rank-constrained approximation problem.*

$$\widehat{W} = \underset{\operatorname{rank} \widehat{W} \leq r}{\arg\min} \|WX_{\text{in}} - \widehat{W}X_{\text{in}}\|_F^2. \tag{1}$$

Let $USV^\top$ be the rank-r SVD on $WX_{\text{in}} \in \mathbb{R}^{d_{\text{out}} \times n}$:

$$USV^\top \approx WX_{\text{in}},$$

where $U \in \mathbb{R}^{d_{\text{out}} \times r}$ is the orthogonal matrix corresponding to the top r left singular vectors. The minimizer to Equation (1) is given by

$$\widehat{W} = UU^\top W.$$

Proof. Denote $Z = WX_{\text{in}}$. By Eckart–Young–Mirsky theorem (Eckart & Young, 1936), we know that the SVD $\widehat{Z} = USV^\top \approx Z$ is the best rank-r approximation to Z, where $U \in \mathbb{R}^{d_{\text{out}} \times r}$, $S = \text{diag}(S_1, \ldots, S_r)$, $V \in \mathbb{R}^{n \times r}$. Furthermore, we have

$$\widehat{Z} = UU^\top Z.$$

Plugging in $Z = WX_{\text{in}}$, we have

$$\widehat{Z} = UU^\top WX_{\text{in}}.$$

Recall that we set $\widehat{W} = UU^\top W$. We see that

$$\|\widehat{Z} - Z\|_F^2 \text{ is minimized } \Rightarrow \|\widehat{W}X_{\text{in}} - WX_{\text{in}}\|_F^2 \text{ is minimized.}$$

Furthermore, as $UU^\top$ is a rank-r projection matrix, we have $\text{rank}(\widehat{W}) \leq r$. Therefore, $\widehat{W}$ is the optimal solution to the rank-constrained minimization problem (Equation (1)). The same reasoning is used in Hsu et al. (2021). □