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
([*]Equal contribution, [†]Equal Advising)

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 without significantly 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

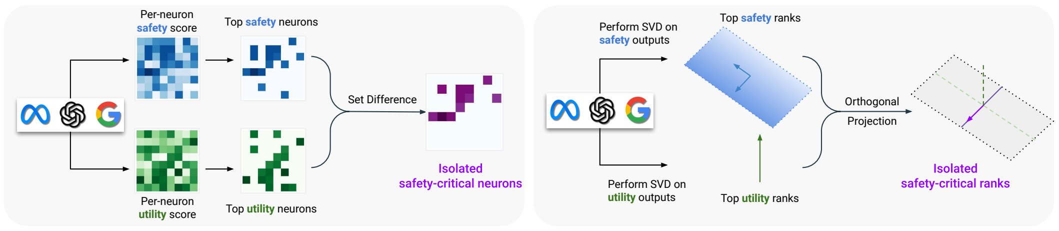
The capabilities of large language models (LLMs) have been significantly improved over the past few years (Brown et al., 2020; OpenAI, 2022; Touvron et al., 2023a; Anthropic, 2023a; Team et al., 2023), yet these models face the challenge of producing inaccurate, misleading, or harmful outputs. Efforts to align them with human values (Ouyang et al., 2022; Bai et al., 2022a;b; Rafailov et al., 2023)are ongoing, yet 'jailbreak' scenarios show that even well-aligned models can be compromised through adversarial prompts (Wei et al., 2023; Carlini et al., 2023; Zou et al., 2023b), persuasion techniques (Zeng et al., 2024), or decoding manipulation (Huang et al., 2024). Furthermore, fine-tuning may also inadvertently weaken a model's safety mechanisms (Qi et al., 2024).

Addressing failure cases in the alignment of LLMs requires a deep understanding of why their safety mechanisms are fragile. Our study aims to contribute to this understanding via *weight attribution* — the process of linking safe behaviors to specific regions within the model's weights. However, a key challenge here is the intricate overlap between *safety* mechanisms and the model's general capabilities, or *utility*. For instance, responsibly handling a harmful instruction entails understanding the instruction, recognizing its harmful intent, and declining it appropriately, which requires a blend of safety awareness and utility capability. We aim to identify minimal safety-critical links within the model that, if disrupted, could compromise its safety without significantly impacting its utility. If there are few such links, it may help explain why safety mechanisms remain brittle.

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 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

Emails: {wby,kaixuanh,yangsibo,thx,xiangyuqi,mengzhou,pmittal,mengdiw,peter.henderson}
@princeton.edu



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.

Table 1: The overview of our weight attribution methods.

	Neuron Attribution	**Rank Attribution**
Output Change	Wanda (Sun et al., 2024), Appendix A.2 $\lvert W\rvert \odot \left(\mathbf{1}\cdot \lVert X_{\text{in}}\rVert_2^\top\right)$	ActSVD $USV^\top \approx WX_{\text{in}},\ \widehat{W} = UU^\top W$
Loss Change	SNIP (Lee et al., 2019), Appendix A.1 $\mathbb{E}_{x\sim D}\lvert W\odot \nabla_W \mathcal{L}(x)\rvert$	N/A
Disentanglement Method	Set difference $S(p,q) = S^s(q) - S^u(p)$	Orthogonal projection $\Delta W(r^u, r^s) = (I - \Pi^u)\Pi^s W$

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.

Safety-critical regions are very sparse in aligned models. We experiment with 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 (§ 3.1). 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 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 (§ 3.1). The attack success rate escalates from 0 to over 90%, yet the model's overall utility remains largely unaffected. Interestingly, excising regions least crucial for safety slightly enhances resistance to jailbreaking attempts – 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, we show that this strategy only offers resistance to minor model modifications (§ 3.3). This suggests that fine-tuning attacks may introduce new pathways that bypass the model's original safety mechanisms. Therefore, further research is needed to develop more robust mitigation strategies.

2 Methodology

To identify and isolate the regions that are exclusively responsible for a model's behaviors, we make modifications to the weights of the model and observe its behavioral change.

We present details of the methods below and summarize them in Table 1.

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 $W \in \mathbb{R}^{d_{\text{out}}\times d_{\text{in}}}$, and a dataset D, SNIP calculates $I(W) = \mathbb{E}_{x\sim D} I(W,x) = \mathbb{E}_{x\sim D}\lvert W\odot \nabla_W \mathcal{L}(x)\rvert$. Appendix A.1 provides more details for derivation. Intuitively, $I(W)$ measures how important each entry is for the behavior of the model on the dataset D. We can attribute the specific behavior of the model to 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 $X_{\text{in}} \in \mathbb{R}^{d_{\text{in}}\times n}$. The Wanda importance score $I(W)$ is given by $I(W) = \lvert W\rvert \odot \left(\mathbf{1}\cdot \lVert X_{\text{in}}\rVert_2^\top\right)$, where we take the row-wise L^2 norm to obtain $\lVert X_{\text{in}}\rVert_2 \in \mathbb{R}^{d_{\text{in}}}$. Appendix A.2 provides more details for derivation.

ActSVD We seek to find a low-rank matrix $\widehat{W}$ such that the Frobenius norm of the change to output is minimized:

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

This can be done by performing SVD on $WX_{\mathrm{in}} \in \mathbb{R}^{d_{\mathrm{out}} \times n}$: $USV^\top \approx WX_{\mathrm{in}}$, where $U \in \mathbb{R}^{d_{\mathrm{out}} \times r}$ 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 F. 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 $\mathrm{rank}(\Delta W) \leq \mathrm{rank}(I - UU^\top) = R - r$, where $R = \mathrm{rank}(W)$. See Appendix B.3 for discussion of related methods.

2.2 Isolating Safety Neurons and Ranks

Assume we have two calibration different datasets: a **utility** dataset D^u and a **safety** dataset D^s. We seek to isolate the safety 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 neurons: Set difference. For the two calibration datasets, assume we obtain two scores I^u and I^s, respectively, either using SNIP or Wanda as in § 2.1. We consider the weight neurons that score least according to I^u but score most according to I^s. 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:$S^u(p) = \{(i,j) \mid I^u_{i,j}$ is the top $p\%$ of $I^u_i\}$. Similarly, $S^s(q) = \{(i,j) \mid I^s_{i,j}$ is the top $q\%$ 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 ranks: Orthogonal projection. Let $R = \mathrm{rank}(W)$. 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 § 2.1, where $\Pi^u = U^u(U^u)^\top, \mathrm{rank}(\Pi^u) = R - r^u$, and $\Pi^s = U^s(U^s)^\top, \mathrm{rank}(\Pi^s) = R - r^s$. To isolate the safety 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

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

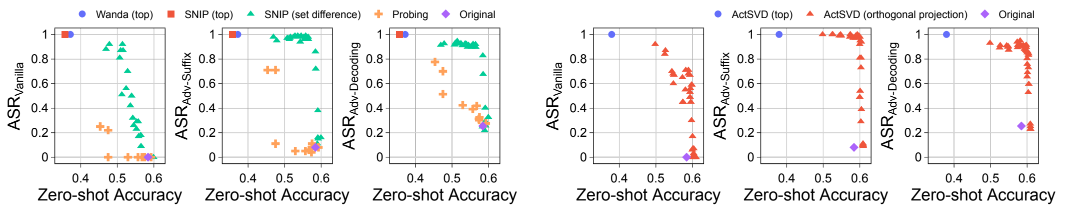
3 Experiments

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

	$\mathrm{ASR}_{\mathrm{Vanilla}}$	$\mathrm{ASR}_{\mathrm{Adv\text{-}Suffix}}$	$\mathrm{ASR}_{\mathrm{Adv\text{-}Decoding}}$
Sample Times	1	1	5
System Prompt	✓	✗	✗
`[INST]`, `[/INST]` wrapper	✓	✓	✗
Adversarial Suffix	✗	✓	✗

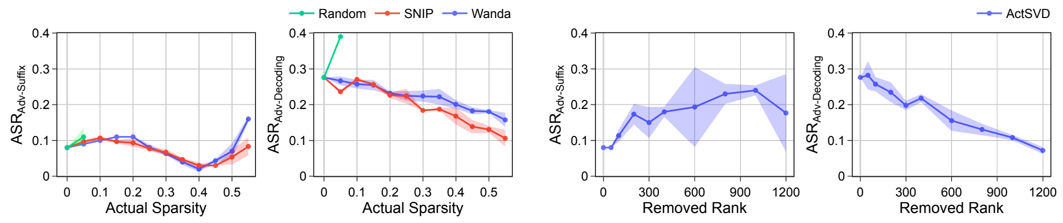
Models and datasets We use `Llama2-7B-chat` and `Llama2-13B-chat` (Touvron et al., 2023b) for their extensive safety tuning process. To identify safety-critical regions in the model, we use two types of datasets: the safety dataset and the utility dataset for attributing safety-related and utility-related behaviors respectively, both in (prompt, response) format. More details are provided in Appendix D.

Measuring utility and safety Following Sun et al. (2024), we measure the model's utility through its averaged zero-shot accuracy of seven tasks from EleutherAI LM Harness (Gao et al., 2023). For safety, we assess the attack success rate (ASR) using `AdvBench`prompts, defining a successful attack when the model's response lacks key patterns indicative of instruction rejection (Zou et al., 2023b). We examine ASR in three contexts: standard ($\mathrm{ASR}_{\mathrm{Vanilla}}$), and two malicious conditions—manipulated decoding ($\mathrm{ASR}_{\mathrm{Adv\text{-}Decoding}}$) and adversarial suffixes ($\mathrm{ASR}_{\mathrm{Adv\text{-}Suffix}}$). Differences in these metrics are detailed in Table 2.



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

Figure 2: ASR and accuracy after removing safety-critical neurons with **sparsity constraint** $< 3\%$ (a) and ranks with **ranks of the weight updates (**$\mathrm{rank}(\Delta W)$**) less than 100** (b) in `Llama2-7B-chat`. 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.



(a) Removing the least Safety-Critical **Neurons** (b) Removing the least Safety-Critical **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-critical ranks identified by ActSVD. Both the importance score and safety-critical 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 $\mathrm{ASR}_{\text{Adv-Suffix}}$ is the adversarial suffixes may not directly transfer to the modified model. See Figure 6 for the results on `Llama2-13B-chat`.

Methods We conduct experiments to identify safety-critical neurons using SNIP (top), Wanda (top), SNIP with set difference, and probing, and safety-critical ranks using ActSVD and ActSVD with orthogonal projection. See Appendix D.1 and Appendix D.2 for more details.

§ 3.1 demonstrates that our set difference and orthogonal projection outperform other methods in isolating safety-critical neurons or ranks. Then in § 3.2, we analyze the overlap between safety and utility, where we observe less overlap in MLP layers than self attention layers. Finally, § 3.3 examines the potential of freezing safety-critical neurons to counter fine-tuning attacks.

3.1 Separating Safety and Utility is Key to Identify Critical Safety Regions

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 § 2.2 to SNIP. Figure 2a presents the Pareto front resulting from set difference-based pruning: Removing $< 3\%$ of neurons pushes ASR close to 1, while maintaining an average zero-shot accuracy above 0.5. Interestingly, if we focus solely on adversarial cases, only pruning less than 1% of neurons can significantly compromise safety while still keeping its accuracy, as shown in Appendix E.4. Similarly, Figure 2b shows that an update of just 2.5% (less than 100 of 4096) ranks significantly increases the model's ASR, while preserving its accuracy. These findings suggest that neurons and ranks for safety are relatively *sparse*.

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 from the original 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 E.2). 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.

Attention head probing scores cannot isolate safety-critical neurons. Probing results in Appendix E.5 suggest that activations of individual attention heads are predictive of identifying harmful versus harmless instructions, which, however, does not necessarily mean the top-scored heads are *solely* responsible for safety generation. To verify this, we prune the top-scored attention heads from

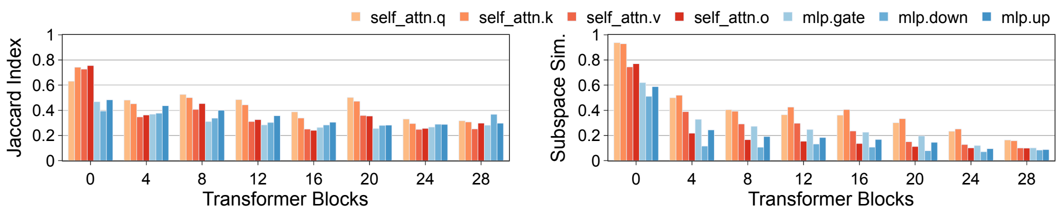


Figure 4: Safety and utility overlapping analysis of each layer inside `Llama2-7B-chat` using: (a) Jaccard Index ($p\% = q\% = 5\%$) and (b) subspace similarity ($\text{rank}(U^u) = \text{rank}(U^s) = 100$). MLP layers show greater differentiation in safety and utility behaviors.

`Llama2-7B-chat`. As shown in Figure 2a, our set difference approach outperforms the probing method consistently, which highlights the need for a disentanglement method.

Pruning least safety-relevant neurons and ranks 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. To verify this, we conduct an ablation study to remove neurons and ranks that are least important for safety from the model. As shown in Figure 3, doing so enhances the robustness of the model against adversarial decoding. More details are available in Appendix E.1.

3.2 MLP Layers Appear to Encode More Differentiated Behaviors

In § 3.1, 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 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 4 shows Jaccard indices across transformer blocks and layers in `Llama2-7B-chat`, using SNIP with $p\%$ and $q\%$ at 5%. Notably, MLP layers exhibit lower Jaccard indices compared to attention layers, suggesting that utility or safety-related knowledge is more differentiated in MLP layers.

Rank-level subspace similarity: We also find that MLP layers appear to encode more differentiated behaviors for safety and utility, from the rank perspective. Specifically, we report 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 4. 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.

3.3 Freezing Safety-Critical Neurons Does Not Stop Fine-Tuning Attacks

We explore whether the identified safety-critical neurons could mitigate the fine-tuning attack (Qi et al., 2024; Yang et al., 2023). We follow Qi et al. (2024)'s fine-tuning attack but 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 $n_{\text{sample}} = 10$ 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 bypass the existing safety mechanisms easily.

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 Alapca examples used for fine-tuning. Freezing safety-critical neurons is insufficient to thwart fine-tuning attacks.

% 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

4 Conclusion

We introduce a pipeline for identifying safety-critical neurons and ranks in aligned LLMs, distinguishing safety from utility and revealing their sparsity—only about 3% and 2.5% at neuron and rank levels, respectively. 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 finding highlights the fragility of safety alignment in current LLMs and suggests using this sparsity as an intrinsic metric to evaluate future models' safety robustness, thereby complementing red teaming efforts. We discuss limitations and future work in Appendix C.

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. 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.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. Fine-grained Analysis of Sentence Embeddings Using Auxiliary Prediction Tasks. In *ICLR*, 2016.

Sajid Ali, Tamer Abuhmed, Shaker El-Sappagh, Khan Muhammad, Jose M Alonso-Moral, Roberto Confalonieri, Riccardo Guidotti, Javier Del Ser, Natalia Díaz-Rodríguez, and Francisco Herrera. 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.

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

Saleh Ashkboos, Maximilian L Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. SliceGPT: Compress Large Language Models by Deleting Rows and Columns. *arXiv preprint arXiv:2401.15024*, 2024.

Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, et al. A General Language Assistant as a Laboratory for Alignment. *arXiv preprint arXiv:2112.00861*, 2021.

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On Pixel-Wise Explanations for Non-linear Classifier Decisions by Layer-Wise Relevance Propagation. *PloS one*, 2015.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a Helpful and Harmless Assistant With Reinforcement Learning From Human Feedback. *arXiv preprint arXiv:2204.05862*, 2022a.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional AI: Harmlessness From AI Feedback. *arXiv preprint arXiv:2212.08073*, 2022b.

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

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language Models are Few-Shot Learners. In *NeurIPS*, 2020.

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

Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. Discovering Latent Knowledge in Language Models Without Supervision. In *ICLR*, 2023.

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

Steven Cao, Victor Sanh, and Alexander M Rush. Low-Complexity Probing via Finding Subnetworks. In *NAACL*, 2021.

Nicholas Carlini, Milad Nasr, Christopher A Choquette-Choo, Matthew Jagielski, Irena Gao, Anas Awadalla, Pang Wei Koh, Daphne Ippolito, Katherine Lee, Florian Tramer, et al. Are Aligned Neural Networks Adversarially Aligned? In *NeurIPS*, 2023.

Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. The Lottery Ticket Hypothesis for Pre-trained BERT Networks. *NeurIPS*, 2020.

Tianyi Chen, Tianyu Ding, Badal Yadav, Ilya Zharkov, and Luming Liang. LoRAShear: Efficient Large Language Model Structured Pruning and Knowledge Recovery. *arXiv preprint arXiv:2310.18356*, 2023.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality, March 2023. URL https://lmsys.org/blog/2023-03-30-vicuna/.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. What does BERT look at? An Analysis of BERT's Attention. In *ACL*, 2019.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. What You Can Cram Into a Single Vector: Probing Sentence Embeddings for Linguistic Properties. In *ACL*, 2018.

Josef Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong Yang. Safe RLHF: Safe Reinforcement Learning from Human Feedback. In *ICLR*, 2024.

Fahim Dalvi, Nadir Durrani, Hassan Sajjad, Yonatan Belinkov, Anthony Bau, and James Glass. What Is One Grain of Sand in the Desert? Analyzing Individual Neurons in Deep NLP Models. In *AAAI*, 2019.

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

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

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

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A Framework for Few-shot Language Model Evaluation, 2023. URL https://zenodo.org/records/10256836.

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. Dissecting Recall of Factual Associations in Auto-Regressive Language Models. In *EMNLP*, 2023.

Demi Guo, Alexander M Rush, and Yoon Kim. Parameter-efficient transfer learning with diff pruning. In *ACL*, 2021.

Wes Gurnee, Theo Horsley, Zifan Carl Guo, Tara Rezaei Kheirkhah, Qinyi Sun, Will Hathaway, Neel Nanda, and Dimitris Bertsimas. Universal Neurons in GPT2 Language Models. *arXiv preprint arXiv:2401.12181*, 2024.

Song Han, Huizi Mao, and William J Dally. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. In *ICLR*, 2016.

Yihui He, Xiangyu Zhang, and Jian Sun. Channel Pruning for Accelerating Very Deep Neural Networks. In *ICCV*, 2017.

John Hewitt and Percy Liang. Designing and Interpreting Probes With Control Tasks. In *EMNLP*, 2019.

Yen-Chang Hsu, Ting Hua, Sungen Chang, Qian Lou, Yilin Shen, and Hongxia Jin. Language Model Compression With Weighted Low-Rank Factorization. In *ICLR*, 2021.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. In *ICLR*, 2022.

Yangsibo Huang, Samyak Gupta, Mengzhou Xia, Kai Li, and Danqi Chen. Catastrophic Jailbreak of Open-source LLMs via Exploiting Generation. In *ICLR*, 2024.

Samyak Jain, Robert Kirk, Ekdeep Singh Lubana, Robert P Dick, Hidenori Tanaka, Edward Grefenstette, Tim Rocktäschel, and David Scott Krueger. Mechanistically Analyzing the Effects of Fine-Tuning on Procedurally Defined Tasks. *arXiv preprint arXiv:2311.12786*, 2023.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient Memory Management for Large Language Model Serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626, 2023.

François Lagunas, Ella Charlaix, Victor Sanh, and Alexander M Rush. Block Pruning For Faster Transformers. In *EMNLP*, 2021.

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

Andrew Lee, Xiaoyan Bai, Itamar Pres, Martin Wattenberg, Jonathan K Kummerfeld, and Rada Mihalcea. A Mechanistic Understanding of Alignment Algorithms: A Case Study on DPO and Toxicity. *arXiv preprint arXiv:2401.01967*, 2024.

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

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning Filters for Efficient ConvNets. In *ICLR*, 2017.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-Time Intervention: Eliciting Truthful Answers from a Language Model. In *NeurIPS*, 2023a.

Yixiao Li, Yifan Yu, Qingru Zhang, Chen Liang, Pengcheng He, Weizhu Chen, and Tuo Zhao. LoSparse: Structured Compression of Large Language Models based on Low-Rank and Sparse Approximation. In *ICML*, 2023b.

Yuhui Li, Fangyun Wei, Jinjing Zhao, Chao Zhang, and Hongyang Zhang. RAIN: Your Language Models Can Align Themselves without Finetuning. In *ICLR*, 2024.

Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. AutoDAN: Generating Stealthy Jailbreak Prompts on Aligned Large Language Models. In *ICLR*, 2024.

Yi Liu, Gelei Deng, Zhengzi Xu, Yuekang Li, Yaowen Zheng, Ying Zhang, Lida Zhao, Tianwei Zhang, and Yang Liu. Jailbreaking ChatGPT via Prompt Engineering: An Empirical Study. *arXiv preprint arXiv:2305.13860*, 2023.

Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning Efficient Convolutional Networks through Network Slimming. In *ICCV*, 2017.

Ekdeep Singh Lubana, Eric J Bigelow, Robert P Dick, David Krueger, and Hidenori Tanaka. Mechanistic Mode Connectivity. In *ICML*, 2023.

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

Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. ThiNet: A Filter Level Pruning Method for Deep Neural Network Compression. In *ICCV*, 2017.

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

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

Pratyush Maini, Michael C Mozer, Hanie Sedghi, Zachary C Lipton, J Zico Kolter, and Chiyuan Zhang. Can Neural Network Memorization Be Localized? In *ICML*, 2023.

Anay Mehrotra, Manolis Zampetakis, Paul Kassianik, Blaine Nelson, Hyrum Anderson, Yaron Singer, and Amin Karbasi. Tree of Attacks: Jailbreaking Black-Box LLMs Automatically. *arXiv preprint arXiv:2312.02119*, 2023.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and Editing Factual Associations in GPT. *NeurIPS*, 2022.

Jack Merullo, Carsten Eickhoff, and Ellie Pavlick. Circuit Component Reuse Across Tasks in Transformer Language Models. In *ICLR*, 2021.

Paul Michel, Omer Levy, and Graham Neubig. Are Sixteen Heads Really Better Than One? In *NeurIPS*, 2019.

Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. 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.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training Language Models to Follow Instructions With Human Feedback. In *NeurIPS*, 2022.

Abhishek Panigrahi, Nikunj Saunshi, Haoyu Zhao, and Sanjeev Arora. Task-Specific Skill Localization in Fine-tuned Language Models. In *ICML*, 2023.

Xiangyu Qi, Yi Zeng, Tinghao Xie, Pin-Yu Chen, Ruoxi Jia, Prateek Mittal, and Peter Henderson. Fine-tuning Aligned Language Models Compromises Safety, Even When Users Do Not Intend To! In *ICLR*, 2024.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language Models are Unsupervised Multitask Learners. *OpenAI blog*, 2019.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS*, 2023.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why Should I Trust You?" Explaining the Predictions of Any Classifier. In *KDD*, 2016.

Victor Sanh, Thomas Wolf, and Alexander Rush. Movement Pruning: Adaptive Sparsity by Fine-Tuning. *NeurIPS*, 2020.

Steffen Schotthöfer, Emanuele Zangrando, Jonas Kusch, Gianluca Ceruti, and Francesco Tudisco. Low-Rank Lottery Tickets: Finding Efficient Low-Rank Neural Networks via Matrix Differential Equations. *NeurIPS*, 2022.

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

Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. "Do Anything Now": Characterizing and Evaluating In-The-Wild Jailbreak Prompts on Large Language Models. *arXiv preprint arXiv:2308.03825*, 2023.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning Important Features Through Propagating Activation Differences. In *ICML*, 2017.

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

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. A Simple and Effective Pruning Approach for Large Language Models. In *ICLR*, 2024.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic Attribution for Deep Networks. In *ICML*, 2017.

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

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: A Family of Highly Capable Multimodal Models. *arXiv preprint arXiv:2312.11805*, 2023.

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

Eric Todd, Millicent L Li, Arnab Sen Sharma, Aaron Mueller, Byron C Wallace, and David Bau. Function Vectors in Large Language Models. *arXiv preprint arXiv:2310.15213*, 2023.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. LLaMA: Open and Efficient Foundation Language Models. *arXiv preprint arXiv:2302.13971*, 2023a.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open Foundation and Fine-Tuned Chat Models. *arXiv preprint arXiv:2307.09288*, 2023b.

Xiaozhi Wang, Kaiyue Wen, Zhengyan Zhang, Lei Hou, Zhiyuan Liu, and Juanzi Li. Finding Skill Neurons in Pre-trained Transformer-based Language Models. In *EMNLP*, 2022.

Ziheng Wang, Jeremy Wohlwend, and Tao Lei. Structured Pruning of Large Language Models. In *EMNLP*, 2020.

Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How Does LLM Safety Training Fail? In *NeurIPS*, 2023.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned Language Models Are Zero-Shot Learners. In *ICLR*, 2022.

Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning Structured Sparsity in Deep Neural Networks. *NeurIPS*, 29, 2016.

Mengzhou Xia, Zexuan Zhong, and Danqi Chen. Structured Pruning Learns Compact and Accurate Models. In *ACL*, 2022.

Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. Sheared LLaMA: Accelerating Language Model Pre-training via Structured Pruning. In *ICLR*, 2024.

Jing Xu, Andrew Lee, Sainbayar Sukhbaatar, and Jason Weston. Some things are more CRINGE than others: Preference Optimization with the Pairwise Cringe Loss. *arXiv preprint arXiv:2312.16682*, 2023.

Kevin Yang, Dan Klein, Asli Celikyilmaz, Nanyun Peng, and Yuandong Tian. RLCD: Reinforcement Learning from Contrastive Distillation for LM Alignment. In *ICLR*, 2024.

Xianjun Yang, Xiao Wang, Qi Zhang, Linda Petzold, William Yang Wang, Xun Zhao, and Dahua Lin. Shadow Alignment: The Ease of Subverting Safely-Aligned Language Models. *arXiv preprint arXiv:2310.02949*, 2023.

Zheng Xin Yong, Cristina Menghini, and Stephen Bach. Low-Resource Languages Jailbreak GPT-4. In *Socially Responsible Language Modelling Research*, 2023.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-Rewarding Language Models. *arXiv preprint arXiv:2401.10020*, 2024a.

Youliang Yuan, Wenxiang Jiao, Wenxuan Wang, Jen-tse Huang, Pinjia He, Shuming Shi, and Zhaopeng Tu. GPT-4 Is Too Smart To Be Safe: Stealthy Chat with LLMs via Cipher. In *ICLR*, 2024b.

Zhihang Yuan, Yuzhang Shang, Yue Song, Qiang Wu, Yan Yan, and Guangyu Sun. ASVD: Activation-aware Singular Value Decomposition for Compressing Large Language Models. *arXiv preprint arXiv:2312.05821*, 2023.

Matthew D Zeiler and Rob Fergus. Visualizing and Understanding Convolutional Networks. In *ECCV*, 2014.

Yi Zeng, Hongpeng Lin, Jingwen Zhang, Diyi Yang, Ruoxi Jia, and Weiyan Shi. How Johnny Can Persuade LLMs to Jailbreak Them: Rethinking Persuasion to Challenge AI Safety by Humanizing LLMs. *arXiv preprint arXiv:2401.06373*, 2024.

Qiusi Zhan, Richard Fang, Rohan Bindu, Akul Gupta, Tatsunori Hashimoto, and Daniel Kang. Removing RLHF Protections in GPT-4 via Fine-Tuning. *arXiv preprint arXiv:2311.05553*, 2023.

Mingyang Zhang, Chunhua Shen, Zhen Yang, Linlin Ou, Xinyi Yu, Bohan Zhuang, et al. Pruning Meets Low-Rank Parameter-Efficient Fine-Tuning. *arXiv preprint arXiv:2305.18403*, 2023a.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. AdaLoRA: Adaptive Budget Allocation for Parameter-Efficient Fine-Tuning. In *ICLR*, 2023b.

Haiyan Zhao, Hanjie Chen, Fan Yang, Ninghao Liu, Huiqi Deng, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, and Mengnan Du. Explainability for Large Language Models: A Survey. *ACM Transactions on Intelligent Systems and Technology*, 2023.

Mengjie Zhao, Tao Lin, Fei Mi, Martin Jaggi, and Hinrich Schütze. Masking as an Efficient Alternative to Finetuning for Pretrained Language Models. In *EMNLP*, 2020.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, et al. Representation Engineering: A Top-Down Approach to AI Transparency. *arXiv preprint arXiv:2310.01405*, 2023a.

Andy Zou, Zifan Wang, J Zico Kolter, and Matt Fredrikson. Universal and Transferable Adversarial Attacks on Aligned Language Models. *arXiv preprint arXiv:2307.15043*, 2023b.

A Neuron-Level Importance Score

A.1 SNIP score

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 SNIP importance score (Lee et al., 2019) 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 $I(W)_{ij}$.

A.2 Wanda score

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):

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

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}.

B Related Work

B.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., 2024; 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.

B.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.

B.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

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

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., $X_{\text{in}} = [X_1, \ldots, X_n] \in \mathbb{R}^{d_{\text{in}} \times n}$.

B.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.

C Limitations and Future Work

We acknowledge several limitations in the current work and identify areas for potential future research. A primary limitation is the limited availability of publicly accessible, safety-aligned models, which constrains our experiments to the Llama2-chat models. It is conceivable that models pre-trained and safety-tuned with different datasets and strategies might demonstrate varying behaviors under our weight attribution pipeline.

In addition, we found that standard attention head probing does not effectively localize safety-critical neurons. It remains possible that advanced probing methods could more successfully achieve this goal. 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 should also examine the potential integration of these methods with our pipeline to enhance weight attribution effectiveness.

Moreover, this 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.

D Experimental Details

D.1 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 (§ 2.1) on safety data as safety-critical. We choose $p = 0.01$.
- **Wanda (top)**: we regard neurons that receive top-$p\%$ Wanda scores (§ 2.1) on safety data as safety-critical. We choose $p = 0.01$.
- **SNIP 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 (§ 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. (2019); 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 D provides more details.

D.2 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 (§ 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 (§ 2.2). We do a grid search for r^u and r^s between 50 and 4000.

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

D.3 Other 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.

Table 4: Typical GPU hours of different experiment types.

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

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 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., "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., 2024) from `Alpaca-Cleaned`, a refined version of the Alpaca dataset (Taori et al., 2023).

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.

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

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

Repeat times To mitigate the potential variability introduced by random seeds, we repeat our experiments on Appendix E.1 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.

https://github.com/gururise/AlpacaDataCleaned

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 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, 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 ASR$_{\text{Adv-Decoding}}$, we configure the sampling temperature to 1.0 when generating responses from `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.

E More Experimental Results

E.1 Pruning least safety-relevant neurons and ranks improves safety.

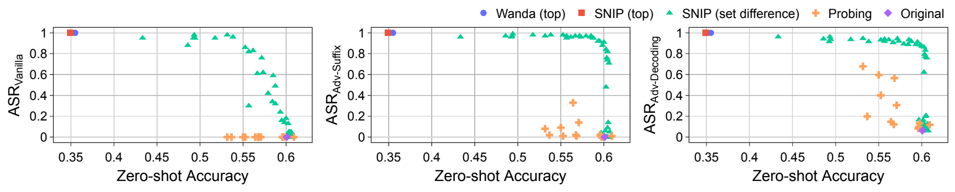
Pruning least safety-relevant neurons improves safety Thinking from the opposite, it's reasonable to hypothesize that the neurons with the lowest safety importance scores could be detrimental for safety. Consequently, eliminating these neurons could potentially enhance the overall safety of the model. 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 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 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.

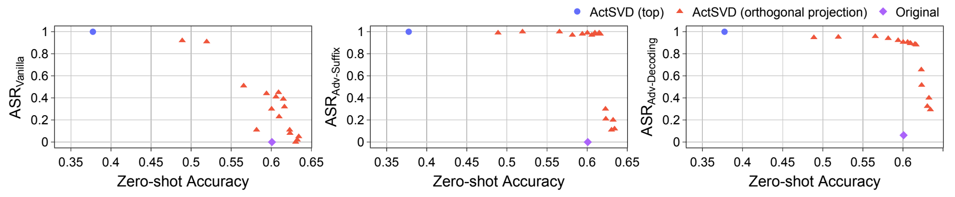
E.2 More results in `Llama2-13B-chat`

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.

According to Zou et al. (2023b), incorporating a broader range of models during training enhances the effectiveness of attacks.

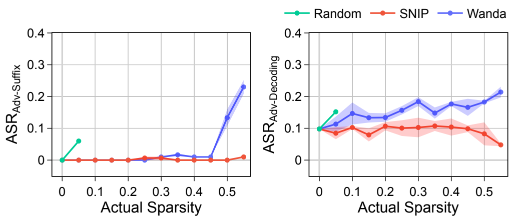


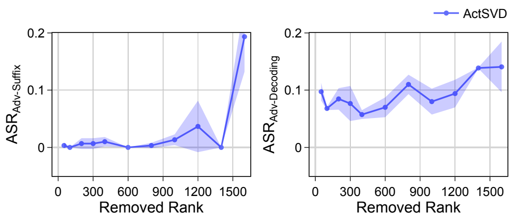
(a) Removing Safety-Critical **Neurons**



(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 Appendix D.1 with **sparsity constraint** $< 3\%$; **(b)** different methods in Appendix D.2 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.





(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.

In accordance with the results on `Llama2-7B-chat` (see § 3), 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 $ASR_{Adv-Suffix}$. 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.

E.3 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.

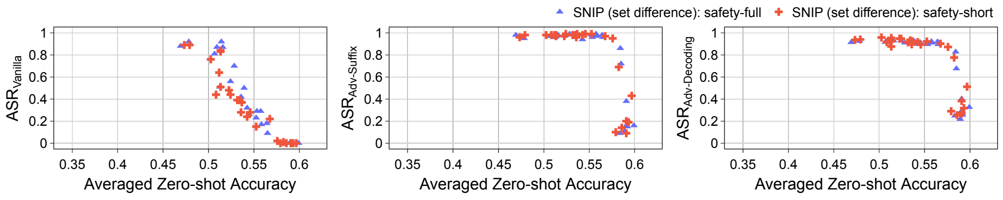


Figure 7: The relationship between ASR and Averaged Zero-shot Accuracy with set difference pruning methods on `Llama2-13B-chat`, using safety and safety-short datasets. The model's performance is very similar in safety and safety-short.

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 Appendix E.1.

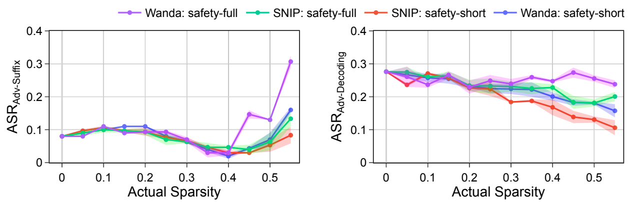


Figure 8: Comparison between using safety-full and safety-short when pruning the least safety-critical neurons on `Llama2-7B-chat`. Compared to safety-full dataset, using safety-short dataset exhibits a more significant change.

E.4 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, Appendix E.4 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 Appendix E.4, 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.

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, Appendix E.4 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.

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.

p	q	**Actual Sparsity**	**ASR$_{\text{Vanilla}}$**	**ASR$_{\text{Adv-Suffix}}$**	**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**	**ASR$_{\text{Vanilla}}$**	**ASR$_{\text{Adv-Suffix}}$**	**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 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.

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

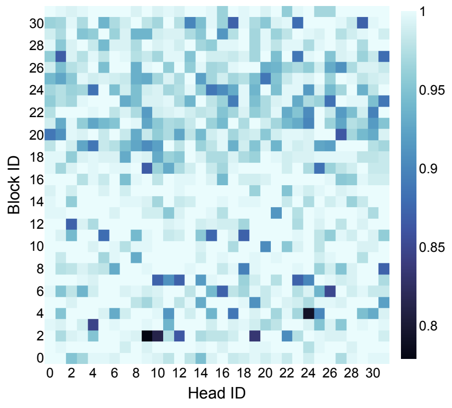
(a) `Llama2-7B-chat` ($R = 4096$)

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

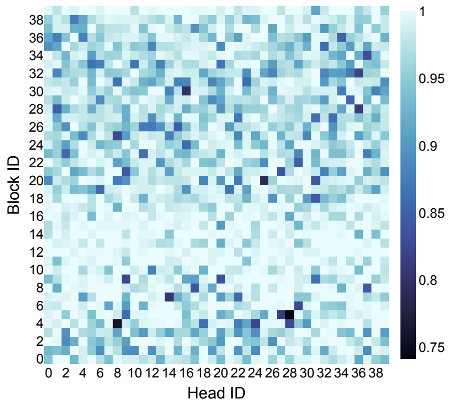
(b) `Llama2-13B-chat` ($R = 5120$)

E.5 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.



(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.

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

F 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.*

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

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). □