LLM-VA: Resolving the Jailbreak-Overrefusal Trade-off via Vector Alignment

Anonymous ACL submission

Abstract

Safety-aligned LLMs suffer from two failure modes: jailbreak (answering harmful inputs) and over-refusal (declining benign queries). Existing vector steering methods adjust the magnitude of answer vectors, but this creates a fundamental trade-off—reducing jailbreak increases over-refusal and vice versa. We identify the root cause: LLMs encode the decision to answer (answer vector v_a) and the judgment of input safety (benign vector v_b) as nearly orthogonal directions, treating them as independent processes. We propose LLM-VA, which aligns v_a with v_b through closed-form weight updates, making the model's willingness to answer causally dependent on its safety assessment—without fine-tuning or architectural changes. Our method identifies vectors at each layer using SVMs, selects safety-relevant layers, and iteratively aligns vectors via minimum-norm weight modifications. Experiments on 12 LLMs demonstrate that LLM-VA achieves 11.45% higher F1 than the best baseline while preserving 95.92% utility, and automatically adapts to each model's safety bias without manual tuning. Code and models are available at `https://anonymous.4open.science/w/LLM-VA-Web-A6C4/`.

1 Introduction

Large language models (LLMs) have achieved remarkable capabilities across diverse NLP tasks (OpenAI, 2024; Team, 2025; AI@Meta, 2024), yet safety alignment remains challenging. Safety-aligned LLMs exhibit two failure modes: *jailbreak*, where the model directly responds to toxic inputs (i.e., queries designed to elicit harmful, unethical, or unsafe responses) (Yi et al., 2024; Zou et al., 2023b; Yuan et al., 2025), and *over-refusal*, where the model unnecessarily declines benign queries (Röttger et al., 2024; Zhang et al., 2025a; Cui et al., 2025). This dual failure mode significantly limits the deployment of LLMs in safety-critical applications, where both reliability and usability are essential. Among approaches to address these issues, vector steering (Zou et al., 2023a; Arditi et al., 2024; Sheng et al., 2025) has gained attention for its efficiency—it manipulates specific directions in the model's latent space without costly retraining, using only simple answer/refuse labels rather than fine-grained annotations.

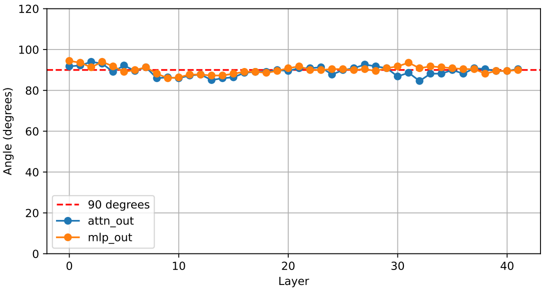


Figure 1: The angles between answer vectors (v_a) and benign vectors (v_b) are approximately 90° across layers in gemma-2-9b-it, indicating near-orthogonality between answer decisions and safety assessments.

However, existing vector steering methods only adjust the *magnitude* of the answer vector, creating a fundamental trade-off: reducing magnitude suppresses jailbreak but increases over-refusal, while amplifying it has the opposite effect (Arditi et al., 2024; Sheng et al., 2025). Recent methods like SCANS (Cao et al., 2025) and CAST (Lee et al., 2024) incorporate input toxicity but require architectural modifications and treat both failure modes as separate objectives (see Table 1). This magnitude-based paradigm cannot fundamentally resolve the trade-off.

We identify the root cause of this trade-off: existing methods control *output behavior* (answer vs. refuse) without considering *input characteristics* (benign vs. toxic). To investigate, we extract two vectors at each layer: the *answer vector* (v_a), indicating whether the model will answer, and the *benign vector* (v_b), indicating whether the input is safe. As shown in Figure 1, these vectors are nearly orthogonal (∼90°) across layers,[1] revealing that LLMs treat answer decisions and safety assessments as *independent* processes. This explains both failure modes: the model may answer toxic inputs (jailbreak) or refuse benign ones (over-refusal) because its willingness to answer is decoupled from its judgment of input safety.

Based on this observation, we propose Large Language Model Vector Alignment (LLM-VA). By aligning these vectors, we make the model's willingness to answer *causally dependent* on its safety assessment (Zou et al., 2023a), rather than treating them as independent decisions. Crucially, LLM-VA achieves this through closed-form weight updates—requiring no gradient-based optimization, fine-tuning, or architectural changes. Our method involves three steps:

- **Vector identification via SVMs**: Train SVMs at each layer to find hyperplanes separating benign/toxic and answer/refuse samples, yielding both v_b and v_a.
- **Layer selection**: Identify layers most relevant to safety decisions based on their contribution to final output and SVM classification accuracy.
- **Vector alignment**: Adjust layer weights to align v_a with v_b, ensuring benign inputs activate the "answer" direction while toxic inputs do not.

Extensive experiments on 12 LLMs demonstrate that LLM-VA achieves 11.45% higher F1 scores (effectiveness on resolving trade-off) than the best baseline (AlphaSteer) (Sheng et al., 2025) with only 4.08% model utility drop, indicating that LLM-VA effectively resolves the jailbreak-overrefusal trade-off while preserving general capabilities. In summary, our contributions are:

- We propose LLM-VA, which, to the best of our knowledge, is the first vector steering method that simultaneously addresses both jailbreak and over-refusal by aligning answer vectors with benign vectors through closed-form weight updates—requiring no gradient-based fine-tuning or architectural changes.
- We demonstrate on 12 LLMs from 5 model families that LLM-VA achieves state-of-the-art safety alignment, and show that it automatically adapts to each model's safety bias—prioritizing jailbreak reduction for vulnerable models and over-refusal reduction for overly conservative ones—without manual tuning.
- We release our code and safety-enhanced weights for 12 LLMs.[2]

2 Related Work

Safety Alignment and the Jailbreak-Overrefusal Trade-off Traditional safety alignment methods—RLHF (Christiano et al., 2017; Stiennon et al., 2020), adversarial training (Xhonneux et al., 2024; Liu et al., 2024a), and rule-based filtering (Zhang et al., 2025b)—require substantial computational resources or lack scalability. Vector steering (Zou et al., 2023a; Arditi et al., 2024) emerged as an efficient alternative, manipulating latent-space directions without retraining. However, these methods create a fundamental trade-off: reducing the answer vector's magnitude suppresses jailbreak but increases over-refusal, while amplifying it has the opposite effect (Arditi et al., 2024; Sheng et al., 2025). This trade-off remains the central unsolved problem in efficient safety alignment.

Vector Steering Methods VectorSteer (Zou et al., 2023a) first identified answer vectors for controlling model outputs through magnitude adjustment. AlphaSteer (Sheng et al., 2025) introduced null-space projection to preserve utility during steering, but remains magnitude-based and thus inherits the trade-off. SCANS (Cao et al., 2025) and CAST (Lee et al., 2024) incorporate input toxicity information, representing progress toward input-aware steering. However, both require architectural modifications (hook layers) and still treat jailbreak and over-refusal as separate objectives to be balanced via hyperparameters. Table 1 summarizes these differences: LLM-VA is the only approach that addresses both failure modes without finetuning or architectural changes.

Internal Representations in LLMs Mechanistic interpretability research reveals that LLMs encode concepts as linear directions in their hidden states (Geva et al., 2021; Elhage et al., 2022; Zou et al., 2023a). Building on this foundation, we discover that answer vectors (v_a) and benign vectors

[1] Results for other LLMs are similar; see Appendix A.

[2] Due to anonymity requirements, we release only Llama3.1-8B-Instruct weights during review. Full weights available at https://figshare.com/s/f2aa365c87a80097a436.

Table 1: Comparison of LLM-VA with other methods on safety alignment and utility preservation.

Method	w/o Finetuning	w/o Model Structure Modification	Over-refusal Mitigation	Jailbreak Mitigation
LLM-VA	✓	✓	✓	✓
Finetuning	✗	✓	✓	✓
VectorSteer	✓	✗	✗	✓
AlphaSteer	✓	✗	✗	✓
CAST	✓	✗	✓	✓
SCANS	✓	✗	✓	✓

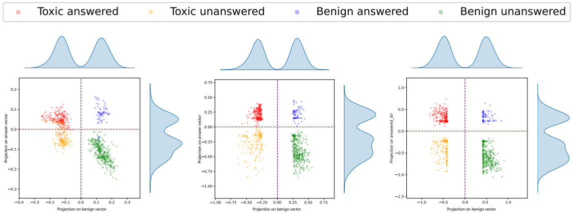


Figure 2: The distributions of the projections onto the benign, answer vectors at different layers of Llama-3.1-8B-Instruct. The left, middle, right figures correspond to the 4th, 16th, and 28th MLP layers, respectively.

(v_b) are nearly orthogonal across layers, explaining why magnitude-based methods cannot resolve the trade-off—they control output behavior independently of input safety. LLM-VA addresses this by aligning these vectors, making the answer decision causally dependent on the safety assessment.

3 Preliminary Analysis

To motivate our approach, we analyze how LLMs internally represent two distinct decisions: (1) whether to answer or refuse a query, and (2) whether the input is benign or toxic.[3] Following Zou et al. (2023a), we extract the answer vector v_a and benign vector v_b at each layer on 128 randomly sampled toxic inputs from S-Eval (Yuan et al., 2025) and 128 benign inputs from ORFuzzSet (Zhang et al., 2025a).[4] We project layer outputs onto these vectors and visualize the distributions in Figure 2. Three key observations emerge:

- **Obs 1: LLMs encode both decisions internally.** Projections onto v_b cleanly separate benign from toxic inputs, while projections onto v_a separate answered from refused samples—both with decision boundaries near zero.
- **Obs 2: Later layers are more discriminative.** Separation quality improves in deeper layers (compare layers 4, 16, and 28 in Figure 2), indicating that later layers are more critical for safety-related decisions.
- **Obs 3: The two decisions are misaligned.** Some toxic inputs project positively onto v_a, while some benign inputs project negatively. This misalignment directly causes jailbreak and over-refusal failures.

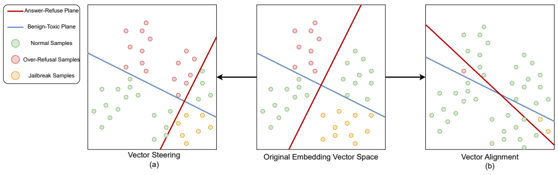


Figure 3: Unlike existing methods that only adjust the magnitude of v_a (trading off jailbreak vs. over-refusal), LLM-VA aligns v_a with v_b to address both issues.

Combined with the near-orthogonality between v_a and v_b (Figure 1), these observations reveal that LLMs treat answer decisions and safety assessments as *independent* processes. We hypothesize that *aligning* v_a with v_b—making the model's willingness to answer depend on its safety judgment—will reduce both failure modes.

Why vector alignment, not magnitude adjustment? Existing vector steering methods (Sheng et al., 2025; Cao et al., 2025; Ray and Bhalani, 2024) only adjust the magnitude of v_a: reducing it decreases jailbreak risk but increases over-refusal, while increasing it has the opposite effect (Figure 3a). In contrast, LLM-VA aligns v_a with v_b (Figure 3b), making the answer decision depend on input safety rather than treating them independently.

Optimization Objective We formalize this goal as maximizing correct response behavior:

$$\max_{\theta} \mathbb{E}_x \Big[\mathbb{I}(y{=}\text{benign}) \cdot \mathbb{I}(f_\theta(x){=}\text{answer}) + \mathbb{I}(y{=}\text{toxic}) \cdot \mathbb{I}(f_\theta(x){=}\text{refuse})\Big] \quad (1)$$

where x is an input, $y \in \{\text{benign}, \text{toxic}\}$ its ground-truth label, and $f_\theta(x) \in \{\text{answer}, \text{refuse}\}$ the model's response. By aligning v_a with v_b, projections onto v_a become correlated with input benignness, optimizing this objective. The following sections detail how LLM-VA achieves this.

4 Methodology

Building on our observation that LLMs encode answer decisions (v_a) and safety assessments

[3] We define "answer" as providing a direct response and "refuse" as declining to respond.

[4] We illustrate with Llama-3.1-8B-Instruct; results are consistent across models.

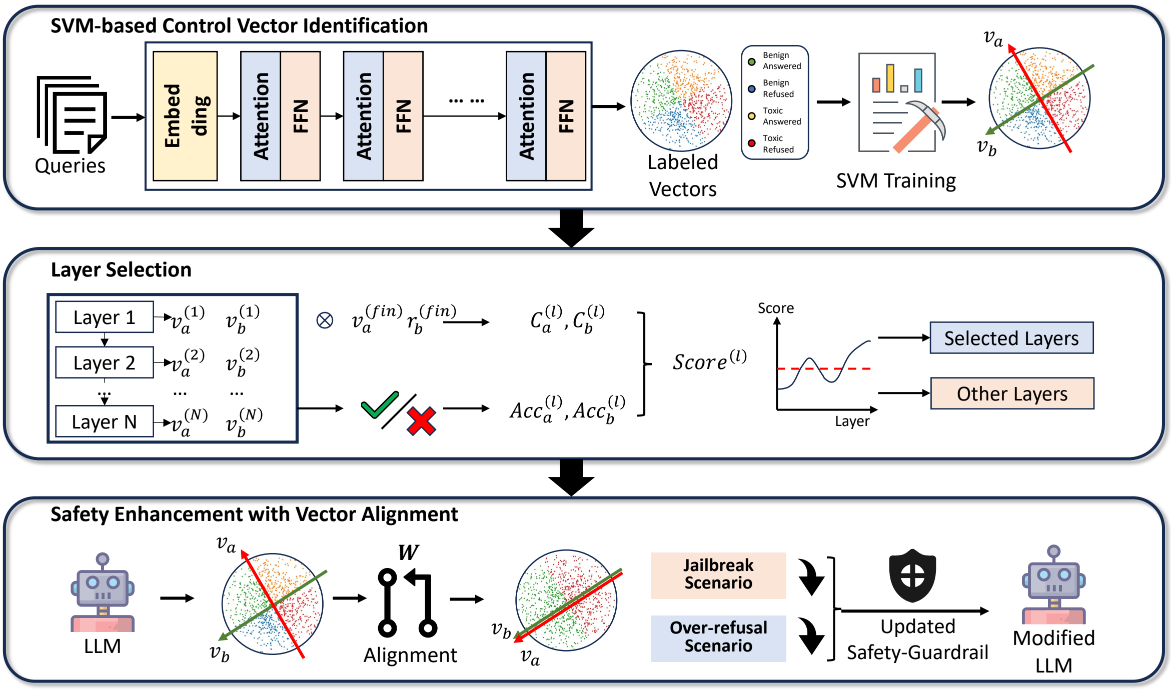


Figure 4: The framework of LLM-VA.

(v_b) as nearly orthogonal directions, we present LLM-VA. Our key insight is that by aligning these vectors through closed-form weight updates—requiring no gradient-based fine-tuning or architectural changes—we can make the model's willingness to answer causally dependent on its safety judgment. As illustrated in Figure 4, LLM-VA mainly consists of three steps: (1) identifying v_a and v_b at each layer via SVMs (Section 4.1), (2) selecting layers most relevant to safety decisions (Section 4.2), and (3) deriving weight update process that aligns these vectors (Section 4.3).

4.1 SVM-based Control Vector Identification

To align vectors at each layer, we must first identify them. Prior work (Zou et al., 2023a; Sheng et al., 2025; Cao et al., 2025) extracts the answer vector from the residual flow at the final layer. However, since the residual flow aggregates contributions from all preceding layers, modifying individual layer weights cannot directly control the final-layer vector. To enable layer-wise weight modification, we instead extract vectors from each layer's output.

At each layer, we train two linear SVMs to find hyperplanes separating (1) benign vs. toxic inputs, and (2) answered vs. refused samples. We use SVMs because they provide interpretable linear decision boundaries: the normal vector of the maximum-margin hyperplane directly yields the control vector, and the margin maximization ensures robustness. The SVMs minimize (Cortes and Vapnik, 1995):

$$\min_{w_{svm},\zeta} \|w_{svm}\|_2^2 + C \sum_{i \in \mathcal{D}} \zeta_i,$$

$$\text{s.t. } y_i(w_{svm} \cdot o_i^{(l)}) \geq 1 - \zeta_i, \ \forall i \in \mathcal{D} \qquad (2)$$

where $o_i^{(l)}$ is the output of layer l for input i, $y_i \in \{-1, 1\}$ is the label ($+1$ for benign/answer, and -1 for toxic/refuse), $C > 0$ is a regularization parameter, and $\zeta_i \geq 0$ are slack variables. We omit the bias term b_{svm} because our empirical analysis shows that decision hyperplanes pass through the origin. This simplifies the subsequent alignment formulation and implementation.

The unit normal vectors of these hyperplanes yield the control vectors:

$$v_b^{(l)} = w_b^{(l)} / \|w_b^{(l)}\| \qquad (3)$$

$$v_a^{(l)} = w_a^{(l)} / \|w_a^{(l)}\| \qquad (4)$$

where $w_b^{(l)}$ and $w_a^{(l)}$ are the SVM weight vectors for benign/toxic and answer/refuse classification at layer l, respectively.

4.2 Layer Selection

Not all layers contribute equally to safety decisions (Geva et al., 2021). Modifying irrelevant layers wastes capacity and may harm utility, so we select layers that are both *influential* (their vectors align with the decisions of final residual stream) and *accurate* (their SVMs reliably distinguish benign/toxic or answer/refuse).[5]

Influence on final decision. Following prior work showing that the residual stream determines final outputs (Zou et al., 2023a; Sheng et al., 2025), we measure how well each layer's vectors align with the vectors of final residual stream:

$$C_a^{(l)} = v_a^{(fin)} \cdot v_a^{(l)}, \quad C_b^{(l)} = v_b^{(fin)} \cdot v_b^{(l)} \qquad (5)$$

High $C^{(l)}$ indicates that modifying layer l's vector direction will propagate to the final decision.

Classification accuracy. We also require that the layer's SVMs accurately separate the two classes. Let $\text{Acc}_a^{(l)}$ and $\text{Acc}_b^{(l)}$ denote validation accuracies for the answer and benign classifiers at layer l.

Combined score. We compute a weighted sum where each term is the product of influence and accuracy for each task:

$$Score^{(l)} = C_a^{(l)} \cdot \text{Acc}_a^{(l)} + C_b^{(l)} \cdot \text{Acc}_b^{(l)} \qquad (6)$$

The multiplicative form within each term ensures we select layers that are *both* influential and accurate for that task—a layer with high influence but low accuracy (or vice versa) contributes little to the score. We select the top L_{select} layers with the highest scores for alignment.

4.3 Vector Alignment

Our goal is to modify each selected layer's weights so that the model's answer decision becomes dependent on its safety assessment. Specifically, for any input, we want the projection onto v_a (which determines answering) to equal the scaled projection onto v_b (which reflects input safety). This ensures benign inputs activate the "answer" direction while toxic inputs suppress it.

Unlike existing methods (Zou et al., 2023a; Sheng et al., 2025; Cao et al., 2025) that insert hook layers and modify the model architecture, we derive a *closed-form* weight update process—requiring no gradient descent or architectural changes. This makes LLM-VA efficient and easy to deploy on standard model-hosting platforms.

Deriving the weight update. For each selected layer, we modify the down-projection matrix W (the matrix that projects from hidden dimension back to model dimension). We seek an update Δ such that (omitting layer indices for clarity):

$$x(W + \Delta)v_a = \frac{\sigma_a}{\sigma_b} x W v_b, \quad \forall x \qquad (7)$$

where σ_a and σ_b are the standard deviations of projections onto v_a and v_b over the training set, respectively. The ratio σ_a/σ_b normalizes for different dynamic ranges of the two directions, ensuring benign inputs (positive v_b projection) produce positive v_a projections and toxic inputs (negative v_b projection) produce negative v_a projections. Rearranging, we require:

$$\Delta v_a = \frac{\sigma_a}{\sigma_b} W v_b - W v_a \qquad (8)$$

The minimum-norm solution (least modification to weights) is given by the pseudoinverse (Penrose, 1955):

$$\Delta^+ = \left(\frac{\sigma_a}{\sigma_b} W v_b - W v_a \right) v_a^T, \qquad (9)$$
$$W' = W + \Delta^+$$

Iterative refinement. A single alignment step may not fully align the vectors because modifying one layer's weights affects the inputs to subsequent layers, causing their effective v_a and v_b directions to shift. We therefore iterate the alignment process T times: in each iteration, we re-extract v_a and v_b from the modified model, recompute layer scores, and apply the weight update. The final model is selected based on validation F1 score. Empirically, most models converge within 20–30 iterations (see Section 5.4).

5 Experiments

We conduct experiments to address the following research questions:

- **RQ1:** How effectively does LLM-VA resolve jailbreak-overrefusal trade-off compared to magnitude-based vector steering methods?
- **RQ2:** How well does LLM-VA preserve model utility?
- **RQ3:** How do key components (vector identification, iteration count, layer selection) affect performance?

[5] Throughout this paper, "layer" refers to either an MLP or attention sublayer unless otherwise specified. Reasons are discussed in Appendix B.

5.1 Experimental Setup

We first describe the experimental settings. Additional details are provided in Appendix D.

Models We conduct experiments on 12 widely-used instruction-tuned LLMs spanning 5 model families, with sizes ranging from 3B to 14B parameters: Llama-3.1 (8B) (AI@Meta, 2024), gemma-2 (9B) (Team, 2024a), Mistral-v0.3 (7B) (Jiang et al., 2023), Phi-3.5 (4B) (Abdin et al., 2024), Phi-4 (4B, 15B) (Microsoft et al., 2025), Qwen2.5 (3B, 7B, 14B) (Team, 2024b; Yang et al., 2024a), and Qwen3 (4B, 8B, 14B) (Team, 2025). This diverse selection allows to evaluate the generalizability of LLM-VA across different architectures and scales.

Datasets For effectiveness evaluation, we use four benchmark datasets: S-Eval-Attack and S-Eval-Risk (Yuan et al., 2025) for jailbreak evaluation, and ORFuzzSet (Zhang et al., 2025a) and Natural Questions (Kwiatkowski et al., 2019) for over-refusal evaluation. To focus on challenging cases, we select 500 samples per dataset where the original models exhibit incorrect behavior (i.e., jailbreak on toxic inputs or over-refusal on benign inputs). Each dataset is split into training, validation, and test sets with a ratio of 8:1:1. For utility preservation, we evaluate on 6 datasets covering diverse NLP tasks including grammar (CoLA (Warstadt et al., 2018)), natural language inference (MNLI (Williams et al., 2018), RTE (Bentivogli et al., 2009)), paraphrase detection (MRPC (Dolan and Brockett, 2005)), sentiment analysis (SST (Socher et al., 2013)), and mathematical reasoning (GSM8K (Cobbe et al., 2021)).[6]

Baselines We compare LLM-VA with several state-of-the-art vector steering methods:

- **VectorSteer** (Zou et al., 2023a): Identifies the answer vector and adjusts its magnitude to control the model's response behavior.
- **AlphaSteer** (Sheng et al., 2025): Extends VectorSteer by introducing null-space projection on representation space to preserve the model's general capabilities while steering.
- **SCANS** (Cao et al., 2025): Dynamically adjusts answer vector magnitude based on input toxicity judgement, using hook layers to incorporate toxicity information.
- **AlphaSteer+**: Our variant of AlphaSteer that uses null-space projection to preserve behavior specifically on correctly-answered samples rather than general capabilities.

Metrics We use attack success rate (ASR) (Zou et al., 2023b) to measure jailbreak vulnerability and over-refusal rate (ORR) (Zhang et al., 2025a) to measure unnecessary refusals. For evaluation of **effectiveness** on resolving the trade-off, we report F1 scores with all the four datasets, where TP=|benign ∩ answered|, FP=|toxic ∩ answered|, FN=|benign ∩ refused|, and TN=|toxic ∩ refused|. For **utility preservation**, we report F1 for classification tasks where TP, FP, FN are defined by the positive class of each task, and accuracy for GSM8K. We employ Qwen3-Guard-Gen-8B (Zhao et al., 2025) as the judge model for evaluating whether responses constitute answers or refusals.[7]

5.2 Effectiveness Results (RQ1)

To evaluate the effectiveness of LLM-VA on jailbreak and over-refusal trade-off, we compare it with magnitude-based vector steering methods across all 12 LLMs. Table 2 presents ASR, ORR, and F1 scores on the test sets.

Overall effectiveness of LLM-VA. LLM-VA achieves an average F1 score of 0.77, representing a 37.02% relative improvement over the original LLMs (0.56). Notably, LLM-VA simultaneously reduces both failure modes: ASR decreases by 18.50% and ORR decreases by 22.00% on average compared to the original LLMs.

Comparison with baselines. LLM-VA outperforms all baselines on 8 of 12 LLMs regarding F1, with a relative improvement of 11.45% over the best baseline (AlphaSteer). VectorSteer, AlphaSteer+ and AlphaSteer, which only adjust answer vector magnitude, show limited improvement on models that already have low ASR but high ORR (e.g., Llama-3.1-8B). SCANS achieves competitive results on some models but requires architectural modifications and shows inconsistent performance across model families.

Adaptive behavior. A key advantage of LLM-VA is its automatic adaptation to each model's initial safety bias. For models with high ASR but low ORR (e.g., Mistral-v0.3-7B with 81% ASR

[6] See Appendix C for dataset details.

[7] See Appendix E for details on judge model selection.

Table 2: Main results of LLM-VA. The best results are **bolded**.

Model	Size	Method	Seval-Aattack ASR↓	Seval-Risk ASR↓	ORFuzzSet ORR↓	NQ ORR↓	Final F1↑	Model	Size	Method	Seval-Aattack ASR↓	Seval-Risk ASR↓	ORFuzzSet ORR↓	NQ ORR↓	Final F1↑
Llama-3.1	8B	Original	12.00%	2.00%	100.00%	**6.00%**	0.6104	Qwen2.5	3B	Original	88.00%	20.00%	62.00%	14.00%	0.5741
		AlphaSteer+	4.00%	**0.00%**	100.00%	10.00%	0.6122			AlphaSteer+	28.00%	**0.00%**	58.00%	10.00%	0.7333
		AlphaSteer	2.00%	**0.00%**	100.00%	**6.00%**	0.6351			AlphaSteer	28.00%	6.00%	62.00%	10.00%	0.7072
		VectorSteer	**0.00%**	**0.00%**	100.00%	12.00%	0.6111			VectorSteer	**22.00%**	2.00%	92.00%	32.00%	0.5067
		SCANS	4.00%	**0.00%**	100.00%	14.00%	0.5931			SCANS	32.00%	4.00%	70.00%	**6.00%**	0.6889
		LLM-VA	14.00%	6.00%	**38.00%**	10.00%	**0.8172**			LLM-VA	44.00%	12.00%	**16.00%**	16.00%	**0.7925**
gemma-2	9B	Original	42.00%	22.00%	98.00%	16.00%	0.4914		7B	Original	86.00%	36.00%	80.00%	4.00%	0.5297
		AlphaSteer+	16.00%	**0.00%**	94.00%	**4.00%**	0.6415			AlphaSteer+	32.00%	**6.00%**	82.00%	**2.00%**	0.6554
		AlphaSteer	16.00%	**0.00%**	98.00%	**4.00%**	0.6242			AlphaSteer	28.00%	8.00%	86.00%	**2.00%**	0.6437
		VectorSteer	10.00%	**0.00%**	98.00%	18.00%	0.5714			VectorSteer	**16.00%**	16.00%	80.00%	24.00%	0.5854
		SCANS	18.00%	12.00%	92.00%	12.00%	0.5890			SCANS	30.00%	18.00%	86.00%	4.00%	0.6145
		LLM-VA	**0.00%**	6.00%	**36.00%**	6.00%	**0.8681**			LLM-VA	54.00%	30.00%	**22.00%**	4.00%	**0.7598**
Mistral-v0.3	7B	Original	88.00%	74.00%	54.00%	4.00%	0.5635		14B	Original	46.00%	22.00%	90.00%	4.00%	0.5668
		AlphaSteer+	28.00%	36.00%	52.00%	2.00%	0.7122			AlphaSteer+	22.00%	4.00%	92.00%	2.00%	0.6386
		AlphaSteer	34.00%	32.00%	46.00%	2.00%	0.7273			AlphaSteer	**2.00%**	4.00%	92.00%	2.00%	0.6795
		VectorSteer	32.00%	28.00%	44.00%	**0.00%**	0.7500			VectorSteer	6.00%	**0.00%**	96.00%	**0.00%**	0.6710
		SCANS	**26.00%**	40.00%	**28.00%**	4.00%	**0.7742**			SCANS	20.00%	24.00%	82.00%	8.00%	0.6215
		LLM-VA	36.00%	**22.00%**	**28.00%**	12.00%	0.7656			LLM-VA	28.00%	22.00%	**66.00%**	2.00%	**0.6911**
Phi-3.5	4B	Original	82.00%	24.00%	90.00%	6.00%	0.5073	Qwen3	4B	Original	84.00%	32.00%	66.00%	**0.00%**	0.5956
		AlphaSteer+	18.00%	2.00%	88.00%	12.00%	0.6250			AlphaSteer+	28.00%	30.00%	48.00%	2.00%	0.7353
		AlphaSteer	26.00%	6.00%	86.00%	10.00%	0.6190			AlphaSteer	34.00%	26.00%	56.00%	**0.00%**	0.7129
		VectorSteer	20.00%	2.00%	78.00%	18.00%	0.6380			VectorSteer	**24.00%**	**2.00%**	48.00%	2.00%	**0.7979**
		SCANS	**4.00%**	**0.00%**	96.00%	40.00%	0.4776			SCANS	28.00%	10.00%	66.00%	2.00%	0.7135
		LLM-VA	66.00%	16.00%	**50.00%**	**4.00%**	**0.6822**			LLM-VA	46.00%	28.00%	**24.00%**	**0.00%**	0.7822
Phi-4	4B	Original	60.00%	16.00%	68.00%	16.00%	0.5918		8B	Original	92.00%	20.00%	72.00%	2.00%	0.5753
		AlphaSteer+	16.00%	8.00%	74.00%	**6.00%**	0.6977			AlphaSteer+	**18.00%**	18.00%	60.00%	10.00%	0.7104
		AlphaSteer	18.00%	**6.00%**	70.00%	**6.00%**	**0.7126**			AlphaSteer	24.00%	14.00%	58.00%	**0.00%**	0.7474
		VectorSteer	20.00%	8.00%	68.00%	**6.00%**	0.7119			VectorSteer	22.00%	14.00%	40.00%	2.00%	0.8020
		SCANS	**14.00%**	12.00%	78.00%	36.00%	0.5513			SCANS	26.00%	**4.00%**	84.00%	10.00%	0.6310
		LLM-VA	70.00%	26.00%	**48.00%**	8.00%	0.6545			LLM-VA	36.00%	8.00%	**24.00%**	**0.00%**	**0.8381**
	15B	Original	22.00%	6.00%	98.00%	**0.00%**	0.6182		14B	Original	86.00%	30.00%	86.00%	**0.00%**	0.5302
		AlphaSteer+	6.00%	**0.00%**	96.00%	2.00%	0.6623			AlphaSteer+	28.00%	32.00%	52.00%	**0.00%**	0.7255
		AlphaSteer	**2.00%**	**0.00%**	96.00%	2.00%	0.6711			AlphaSteer	26.00%	10.00%	**46.00%**	**0.00%**	**0.7897**
		VectorSteer	**2.00%**	2.00%	94.00%	4.00%	0.6667			VectorSteer	**18.00%**	**0.00%**	72.00%	**0.00%**	0.7399
		SCANS	12.00%	**0.00%**	94.00%	6.00%	0.6410			SCANS	30.00%	18.00%	82.00%	40.00%	0.4785
		LLM-VA	12.00%	6.00%	**38.00%**	**0.00%**	**0.8526**			LLM-VA	46.00%	14.00%	56.00%	**0.00%**	0.7129

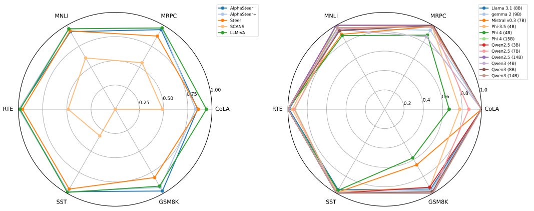


Figure 5: Left: Average utility preservation by method. Right: Utility preservation per LLM with LLM-VA. Values near 1.0 indicate minimal degradation.

and 29% ORR), LLM-VA primarily reduces ASR to ensure safety. Conversely, for models with low ASR but high ORR (e.g., Llama-3.1-8B with 7% ASR and 53% ORR), it primarily decreases ORR to enhance usability. This adaptive behavior emerges naturally from vector alignment without manual hyperparameter tuning for different models.

Cases requiring further analysis. Four models (Phi-3.5-4B, Phi-4-4B, Mistral-v0.3-7B, and Qwen3-14B) do not achieve the highest F1 with LLM-VA. We analyze these cases in Section 5.4 and show that the suboptimal performance stems from iteration count sensitivity rather than fundamental limitations of the approach.

5.3 Utility Preservation Results (RQ2)

Besides effectiveness in resolving trade-off, we also evaluate model utility preservation on 6 benchmark datasets covering classification and mathematical reasoning tasks. Figure 5 shows the results across methods and models.

Overall utility preservation. LLM-VA preserves 95.92% of the original model's utility on average, outperforming all baseline methods. For 9 of 12 LLMs, utility preservation exceeds 95%, demonstrating that LLM-VA successfully enhances alignment without sacrificing general capabilities.

Comparison with baselines. SCANS shows the largest utility degradation (averaging 40.98%) because aggressive magnitude adjustments disrupt the model's internal representations. VectorSteer performs better (89.74%) but still falls short of LLM-VA due to its architectural modifications. AlphaSteer and AlphaSteer+ achieve competitive preservation (94.50% and 94.48%) through null-space projection, but LLM-VA still outperforms them while achieving substantially better alignment.

Task-specific analysis. The utility impact varies across task types. Classification tasks (COLA, MNLI, RTE, MRPC, SST) show minimal degradation, with most models preserving over 97% performance. Mathematical reasoning (GSM8K) is more affected, with 91.60% average preservation. This is expected because math reasoning requires precise logical chains that can be disrupted by representation changes. Nevertheless, the impact remains limited compared to the alignment gains.

Model size effects. Larger and more capable LLMs demonstrate better utility preservation. The three models with lowest preservation—Phi-3.5-4B (92.1%), Phi-4-4B (91.8%), and Mistral-v0.3-7B

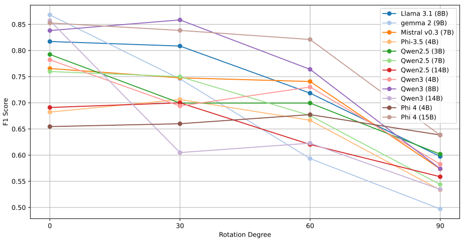
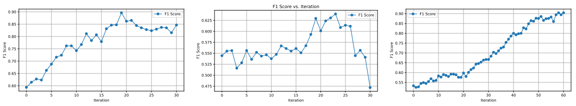


Figure 6: F1 scores with randomly distorted vectors at different angles D from the original benign and answer vectors.



(a) Llama 3.1 (8B) (b) Phi 4 (4B) (c) Qwen 3 (14B)

Figure 7: F1 scores vs. iteration number T for three representative models.

(93.2%)—are either among the smallest models or have documented limitations in benchmarks (Fourrier et al., 2024; Gao et al., 2021). This suggests that larger models have more robust internal representations that better tolerate the weight modifications introduced by vector alignment.

5.4 Ablation Studies (RQ3)

We analyze three key components: vector identification accuracy, iteration count, and layer selection.

Vector Identification. To validate our SVM-based vector identification, we replace v_a and v_b with random vectors D degrees away from the originals, where D ranges from 30° to 90° (Figure 6). The performance degradation correlates with distortion angle: at $D = 90°$ (orthogonal to the true vectors), F1 drops by 24.82% on average, and all 12 models underperform. At $D = 60°$, all models still show degradation. However, at $D = 30°$, F1 only drops by 5.40%, indicating that LLM-VA is robust to small inaccuracies—a practical advantage since SVM hyperplanes may not perfectly capture true decision boundaries—while confirming that accurate identification remains essential.

Iteration Number. We vary iteration count T from 1 to 30 (Figure 7). For clarity, we show the results of three representative models and put the full results in Appendix F. Models exhibit distinct convergence patterns: Llama 3.1 (8B) shows rapid improvement and stabilizes around $T = 19$; Phi 4 (4B) peaks at $T = 19$ but then degrades with additional iterations, suggesting over-modification; Qwen 3 (14B) continues improving through $T = 30$ and beyond (as shown in Figure 7, we extended to $T = 60$ and observed continued gains).

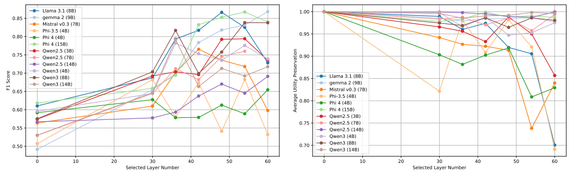


Figure 8: Impact of L_{select} on F1 (left) and utility (right).

These patterns explain the suboptimal results in Table 2: Mistral-v0.3-7B, Phi-3.5-4B and Phi-4-4B suffer from over-modification (smaller and performance-limited models (Fourrier et al., 2024; Gao et al., 2021) are more susceptible to over-modification), while Qwen3-14B underperforms due to under-iteration. This suggests that model-specific iteration tuning or early stopping based on validation performance is important.

Layer Selection. Figure 8 shows F1 and utility as L_{select} varies from 30 to 60. For alignment, most models exhibit a non-monotonic trend with an optimal L_{select}: too few layers limit effectiveness, while too many cause overfitting. For utility preservation, most models remain stable until L_{select} exceeds a threshold, at which point early layers are modified and utility drops sharply. This confirms that later layers are more relevant to safety decisions while early layers are critical for general capabilities, motivating our contribution-score-based layer selection (Section 4.2).

6 Conclusion

In this work, we presented LLM-VA, a novel approach that simultaneously addresses jailbreak and over-refusal by aligning the answer vector with the benign vector through closed-form weight updates—making the model's willingness to answer causally dependent on its safety judgment without requiring fine-tuning or architectural changes. Experiments on 12 widely used LLMs from 5 model families demonstrate a 11.45% F1 improvement over the best baseline while preserving 95.92% utility, and our ablation studies confirm the importance of accurate vector identification and model-specific hyperparameter tuning.

7 Limitations

Binary toxicity assumption. We consider only binary classification (benign vs. toxic), whereas real-world toxicity is nuanced and multi-dimensional. Extending LLM-VA to multi-class or fine-grained toxicity classification remains future work.

Model scale. Our experiments cover models from 3B to 14B parameters. The effectiveness of LLM-VA on larger models (e.g., 70B+) remains to be validated, as these models may have different internal representations and require different hyperparameter settings.

Training data dependency. LLM-VA requires labeled benign/toxic samples to train the SVMs for vector identification. The quality and representativeness of this training data directly affect alignment performance, and obtaining such labels may not always be straightforward.

Reasoning models. Vector steering methods, including LLM-VA, are difficult to apply to LLMs with chain-of-thought reasoning. The control vectors must be identified after reasoning steps are generated, which is computationally expensive, and the randomness in reasoning makes accurate vector identification challenging.

Model-specific tuning. As shown in our ablation studies, optimal iteration count and layer selection vary across models. While LLM-VA uses validation-based selection, this requires tuning for a new model, limiting plug-and-play applicability.

Transferability. The performance of the existing vector steering methods, including LLM-VA, on unseen datasets varies depending on tasks and models (Appendix G). This imply that current steering methods may need to treat different tasks or domains separately, and improving transferability remains future work.

Static alignment. The alignment is performed once and does not adapt to new threats or evolving definitions of harmful content. Periodic re-alignment may be needed as the threat landscape changes.

Customized Trade-off. LLM-VA aims to improve both jailbreak and over-refusal behavior simultaneously. However, in certain applications (e.g., healthcare (Al-Garadi et al., 2025; Yang et al., 2024b) or PLC code generation (Liu et al., 2024b)), users may prefer to prioritize one aspect over the other. Extending LLM-VA to allow for customizable trade-offs remains future work.

Experimental methodology. Our results are based on single runs with a fixed random seed. While we observe consistent improvements across 12 models, incorporating statistical significance tests would further strengthen our empirical findings.

8 Ethical Considerations

8.1 Potential Risks

Though LLM-VA aims to enhance the safety alignment of LLMs, it can be misused to manipulate model behaviors in unintended ways. For instance, attackers could potentially exploit the vector alignment technique to bypass safety mechanisms or introduce harmful biases into the model. Besides, the datasets used for training and evaluation may contain biases.

8.2 AI Assistants Usage

We employ GPT-5.2 (OpenAI, 2024) and Github Copilot[8] to assist in writing code for experiments. We carefully review and verify all AI-generated content to ensure accuracy and integrity.

References

Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Qin Cai, Vishrav Chaudhary, Dong Chen, Dongdong Chen, and 110 others. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *Preprint*, arXiv:2404.14219.

AI@Meta. 2024. Llama 3 model card.

Mohammed Al-Garadi, Tushar Mungle, Abdulaziz Ahmed, Abeed Sarker, Zhuqi Miao, and Michael E. Matheny. 2025. Large language models in healthcare. *Preprint*, arXiv:2503.04748.

Andy Arditi, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. 2024. Refusal in language models is mediated by a single direction. *Advances in Neural Information Processing Systems*, 37:136037–136083.

[8] `https://github.com/features/copilot`

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2009. The fifth pascal recognizing textual entailment challenge. *TAC*, 7(8):1.

Zouying Cao, Yifei Yang, and Hai Zhao. 2025. Scans: Mitigating the exaggerated safety for llms via safety-conscious activation steering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 23523–23531.

Jianfeng Chi, Ujjwal Karn, Hongyuan Zhan, Eric Smith, Javier Rando, Yiming Zhang, Kate Plawiak, Zacharie Delpierre Coudert, Kartikeya Upasani, and Mahesh Pasupuleti. 2024. Llama guard 3 vision: Safeguarding human-ai image understanding conversations. *arXiv preprint arXiv:2411.10414*.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20(3):273–297.

Justin Cui, Wei-Lin Chiang, Ion Stoica, and Cho-Jui Hsieh. 2025. Or-bench: An over-refusal benchmark for large language models. *Preprint*, arXiv:2405.20947.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the International Workshop on Paraphrasing*.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, and 1 others. 2022. Toy models of superposition. *arXiv preprint arXiv:2209.10652*.

Clémentine Fourrier, Nathan Habib, Alina Lozovskaya, Konrad Szafer, and Thomas Wolf. 2024. Open llm leaderboard v2. `https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard`.

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, Jason Phang, Laria Reynolds, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2021. A framework for few-shot language model evaluation.

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

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, and 1 others. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Bruce W Lee, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Erik Miehling, Pierre Dognin, Manish Nagireddy, and Amit Dhurandhar. 2024. Programming refusal with conditional activation steering. *arXiv preprint arXiv:2409.05907*.

Fan Liu, Zhao Xu, and Hao Liu. 2024a. Adversarial tuning: Defending against jailbreak attacks for llms. *arXiv preprint arXiv:2406.06622*.

Zihan Liu, Ruinan Zeng, Dongxia Wang, Gengyun Peng, Jingyi Wang, Qiang Liu, Peiyu Liu, and Wenhai Wang. 2024b. Agents4plc: Automating closed-loop plc code generation and verification in industrial control systems using llm-based agents. *arXiv preprint arXiv:2410.14209*.

Microsoft, :, Abdelrahman Abouelenin, Atabak Ashfaq, Adam Atkinson, Hany Awadalla, Nguyen Bach, Jianmin Bao, Alon Benhaim, Martin Cai, Vishrav Chaudhary, Congcong Chen, Dong Chen, Dongdong Chen, Junkun Chen, Weizhu Chen, Yen-Chun Chen, Yi ling Chen, Qi Dai, and 57 others. 2025. Phi-4-mini technical report: Compact yet powerful multimodal language models via mixture-of-loras. *Preprint*, arXiv:2503.01743.

OpenAI. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, and 1 others. 2011. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

R. Penrose. 1955. A generalized inverse for matrices. *Mathematical Proceedings of the Cambridge Philosophical Society*, 51(3):406–413.

Ruchira Ray and Ruchi Bhalani. 2024. Mitigating exaggerated safety in large language models. *arXiv preprint arXiv:2405.05418*.

Paul Röttger, Hannah Rose Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy. 2024. Xstest: A test suite for identifying exaggerated safety behaviours in large language models. *Preprint*, arXiv:2308.01263.

Leheng Sheng, Changshuo Shen, Weixiang Zhao, Junfeng Fang, Xiaohao Liu, Zhenkai Liang, Xiang Wang, An Zhang, and Tat-Seng Chua. 2025. Alphasteer: Learning refusal steering with principled null-space constraint. *arXiv preprint arXiv:2506.07022*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in neural information processing systems*, 33:3008–3021.

Gemma Team. 2024a. Gemma.

Qwen Team. 2024b. Qwen2.5: A party of foundation models.

Qwen Team. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2018. Neural network acceptability judgments. *arXiv preprint 1805.12471*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long papers)*, pages 1112–1122.

Sophie Xhonneux, Alessandro Sordoni, Stephan Günnemann, Gauthier Gidel, and Leo Schwinn. 2024. Efficient adversarial training in llms with continuous attacks. *Advances in Neural Information Processing Systems*, 37:1502–1530.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and 40 others. 2024a. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Yifan Yang, Qiao Jin, Furong Huang, and Zhiyong Lu. 2024b. Adversarial attacks on large language models in medicine. *Preprint*, arXiv:2406.12259.

Sibo Yi, Yule Liu, Zhen Sun, Tianshuo Cong, Xinlei He, Jiaxing Song, Ke Xu, and Qi Li. 2024. Jailbreak attacks and defenses against large language models: A survey. *Preprint*, arXiv:2407.04295.

Xiaohan Yuan, Jinfeng Li, Dongxia Wang, Yuefeng Chen, Xiaofeng Mao, Longtao Huang, Jialuo Chen, Hui Xue, Xiaoxia Liu, Wenhai Wang, Kui Ren, and Jingyi Wang. 2025. S-eval: Towards automated and comprehensive safety evaluation for large language models. *Proceedings of the ACM on Software Engineering*, 2(ISSTA):2136–2157.

Haonan Zhang, Dongxia Wang, Yi Liu, Kexin Chen, Jiashui Wang, Xinlei Ying, Long Liu, and Wenhai Wang. 2025a. Orfuzz: Fuzzing the "other side" of llm safety – testing over-refusal. *Preprint*, arXiv:2508.11222.

Shenyi Zhang, Yuchen Zhai, Keyan Guo, Hongxin Hu, Shengnan Guo, Zheng Fang, Lingchen Zhao, Chao Shen, Cong Wang, and Qian Wang. 2025b. Jbshield: Defending large language models from jailbreak attacks through activated concept analysis and manipulation. *arXiv preprint arXiv:2502.07557*.

Haiquan Zhao, Chenhan Yuan, Fei Huang, Xiaomeng Hu, Yichang Zhang, An Yang, Bowen Yu, Dayiheng Liu, Jingren Zhou, Junyang Lin, and 1 others. 2025. Qwen3guard technical report. *arXiv preprint arXiv:2510.14276*.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, and 1 others. 2023a. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*.

Andy Zou, Zifan Wang, J. Zico Kolter, and Matt Fredrikson. 2023b. Universal and transferable adversarial attacks on aligned language models. *Preprint*, arXiv:2307.15043.

A Angles between Answer Vectors and Benign Vectors

As shown in Figure 9, the angles between the answer vectors and benign vectors of different LLMs are approximately 90°, indicating that they are nearly orthogonal.

B Discussion on Layer Type Selection

In Section 4.2, we mention that we treat both MLP and attention sublayers as "layers" for selection. This is because both types of sublayers contribute to the model's internal representations and decision-making processes. Modifying either type can influence the model's behavior regarding safety alignment. Besides, we conduct preliminary experiments to compare the combined score (Eq. 6) distributions of MLP and attention sublayers. The results are presented in Figure 10. The results show that both MLP and attention sublayers exhibit similar contribution score distributions across different LLMs. Later layers tend to have higher contribution scores, indicating their greater relevance to safety-related decisions. Therefore, we treat both MLP and attention sublayers equally in our layer selection process.

C Additional Instructions on General Ability Datasets

In this section, we provide detailed instructions on the general ability datasets used in our experiments.

- **Corpus of Linguistic Acceptability (COLA)** (Warstadt et al., 2018) is a dataset for evaluating the grammatical acceptability of sentences. Each sample consists of a sentence and a binary label indicating whether the sentence is grammatically acceptable or not.

- **Multi-Genre Natural Language Inference (MNLI)** (Williams et al., 2018) is a large-scale dataset for natural language inference. Each sample consists of a pair of sentences annotated with textual entailment labels.

- **Recognizing Textual Entailment (RTE)** (Bentivogli et al., 2009) is a dataset for evaluating the ability of models to recognize textual entailment. Each sample consists of a pair of sentences where one sentence is the premise and the other is the hypothesis.

- **Microsoft Research Paraphrase Corpus (MRPC)** (Dolan and Brockett, 2005) is a dataset for evaluating the ability of models to recognize paraphrases. Each sample consists of a pair of sentences extracted from online news sources, with human annotations indicating whether each pair is semantically equivalent or not.

- **Stanford Sentiment Treebank (SST)** (Socher et al., 2013) is a dataset for sentiment analysis. Each sample consists of a sentence and a binary label indicating whether the sentiment of the sentence is positive or negative.

- **GSM8K** (Cobbe et al., 2021) is a dataset for evaluating the mathematical reasoning ability of models. Each sample consists of a math word problem and its corresponding solution.

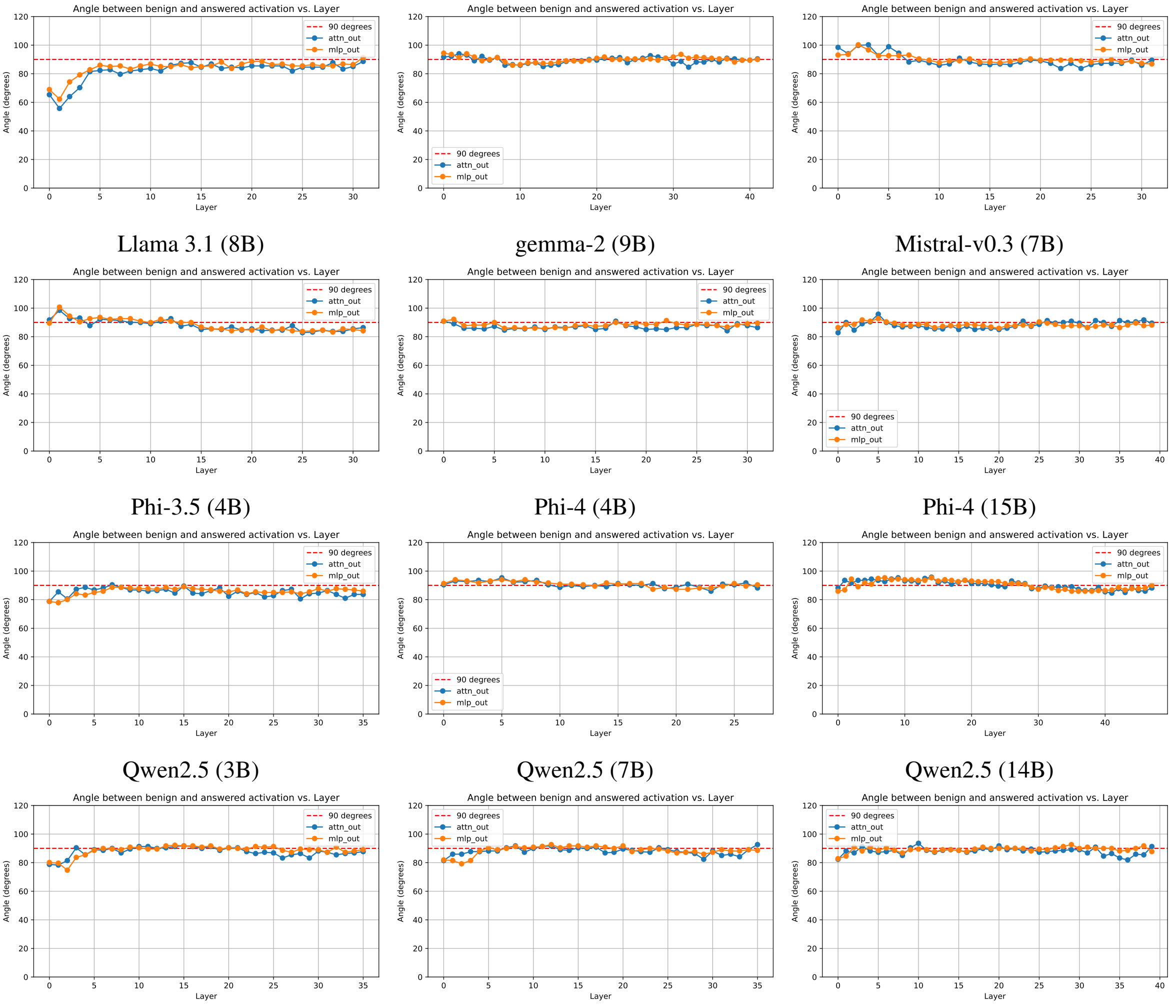
D Details about Experimental Setup

We implement LLM-VA with max iteration number $T = 30$. The final modified model is obtained by selecting the best model on the validation set during the iterations. The numbers of selected layers L_{select} of each model are shown in Table 3. For the SVM-based vector identification, we use the default regularization parameter $C = 1.0$ from scikit-learn (Pedregosa et al., 2011). For baseline methods, we follow the original papers and use the default hyperparameters. If the original papers do not provide hyperparameter settings for certain models, we transfer the hyperparameters from similar models (e.g., models with the same architecture or in the same family). All experiments are conducted on 2×80 GB A100 GPUs. We use the default generation configurations in Hugging Face Transformers[9] for base LLMs during inference. The temperature parameters of all models are set to 0.0 to ensure deterministic outputs. We use a fixed random seed of 42 for reproducibility across all experiments.

E Details on Judge Model Selection

As far as we know, Qwen3-Guard-Gen-8B (Zhao et al., 2025) is currently the only open-source LLM specifically designed to evaluate jailbreak and over-refusal behaviors. We also considered combining multiple judge models to realize the evaluation (e.g., LlamaGuard 3 (Chi et al., 2024) for jailbreak and OR-Judge (Zhang et al., 2025a) for over-

[9] https://huggingface.co/docs/transformers/index



Llama 3.1 (8B)

gemma-2 (9B)

Mistral-v0.3 (7B)

Phi-3.5 (4B)

Phi-4 (4B)

Phi-4 (15B)

Qwen2.5 (3B)

Qwen2.5 (7B)

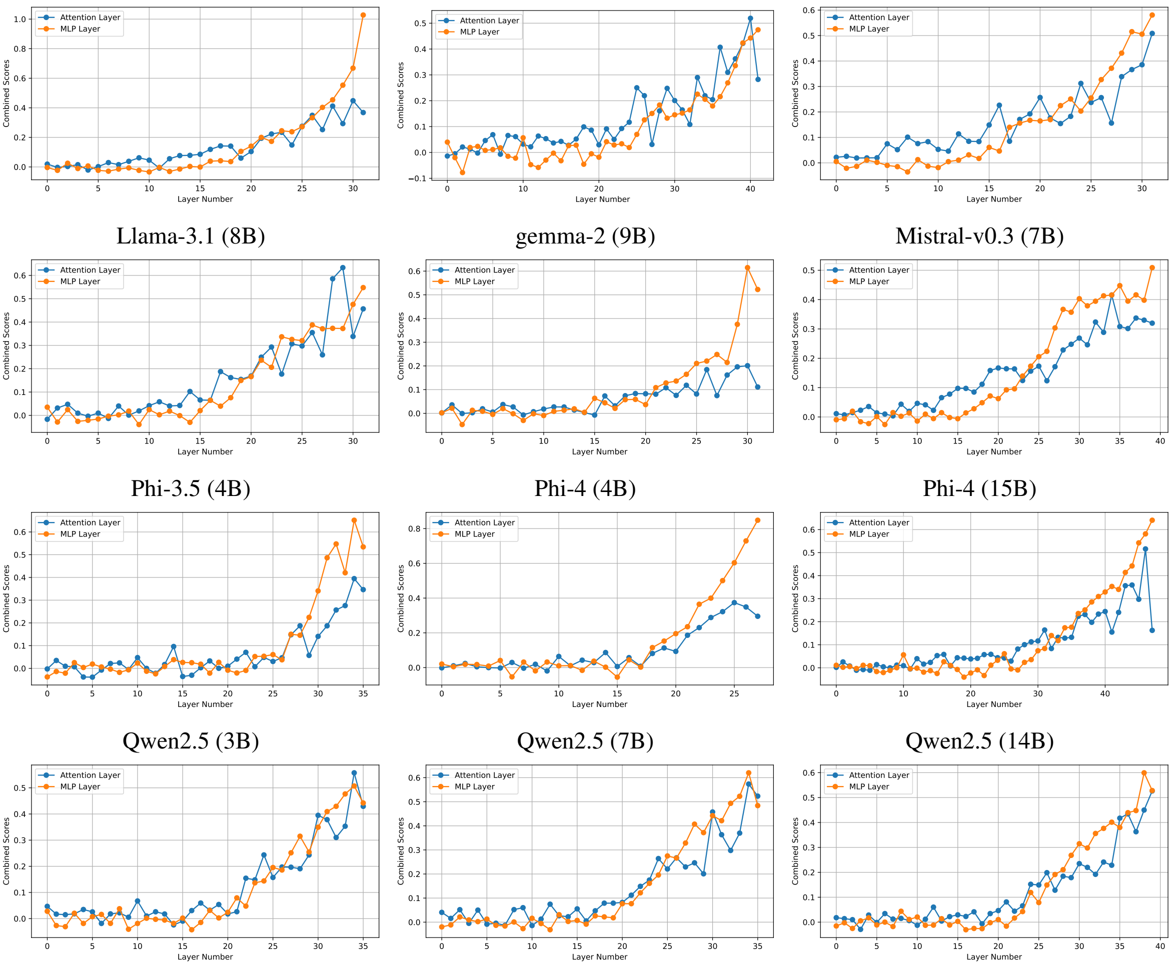
Qwen2.5 (14B)

Qwen3 (4B)

Qwen3 (8B)

Qwen3 (14B)

Figure 9: Angles between answer vectors and benign vectors of different LLMs.



Llama-3.1 (8B)

gemma-2 (9B)

Mistral-v0.3 (7B)

Phi-3.5 (4B)

Phi-4 (4B)

Phi-4 (15B)

Qwen2.5 (3B)

Qwen2.5 (7B)

Qwen2.5 (14B)

Qwen3 (4B)

Qwen3 (8B)

Qwen3 (14B)

Figure 10: Comparison of combined scores between MLP and attention sublayers across different LLMs.

Table 3: Selected layer numbers of different models in LLM-VA.

Model	Llama-3.1	Gemma-2	Mistral-v0.3	Phi-3.5	Phi-4		Qwen2.5			Qwen3		
Size	8B	9B	7B	4B	4B	15B	3B	7B	14B	4B	8B	14B
# Selected Layers	42	60	42	30	48	54	60	36	54	48	60	48

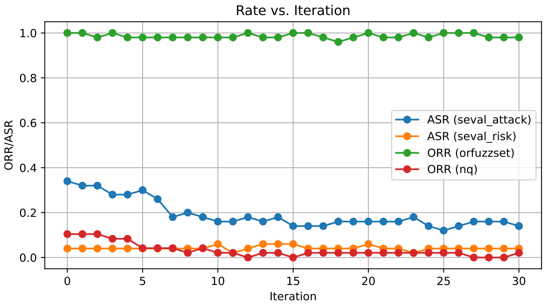


Figure 11: An example of evaluation results with combined judge models.

refusal). However, due to their different judgement criteria, combining multiple judge models may lead to inconsistent evaluations. As a result, LLM-VA will find incorrect vectors to align, leading to suboptimal performance. Figure 11 shows an example of such inconsistent evaluations. The ORR evaluated by OR-Judge reaches 100% due to the inconsistency between the two judge models. Therefore, we choose Qwen3-Guard-Gen-8B as the sole judge model for a consistent evaluation of both jailbreak and over-refusal behaviors.

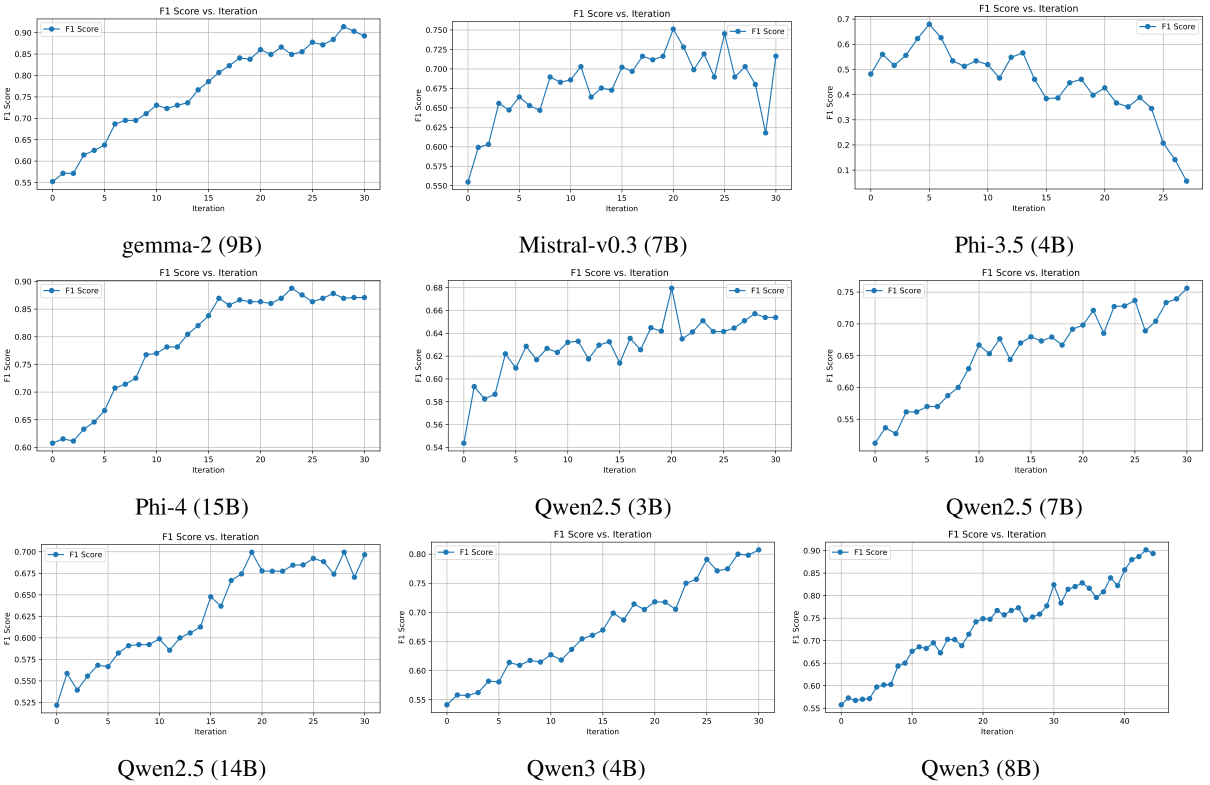
F Detailed Results on Iteration Number

The detailed results on the impact of iteration number of each LLM are shown in Figure 12.

G Transferability Experiments

To evaluate the transferability of LLM-VA, we assess how well the vector alignment learned on the training datasets generalizes to unseen datasets. We evaluate the modified models on three additional jailbreak datasets (XSTest-Toxic (Röttger et al., 2024), OR-Bench-Toxic (Cui et al., 2025), and AdvBench (Zou et al., 2023b)) and two over-refusal datasets (XSTest (Röttger et al., 2024) and OR-Bench (Cui et al., 2025)) that are not included in the training set. The results are shown in Table 4.

The results show that the performance of LLM-VA on unseen datasets varies across different models. While LLM-VA maintains reasonable safety alignment on most unseen datasets, the performance degradation compared to the training datasets indicates that further research is needed to improve the generalization of vector steering methods.



gemma-2 (9B) Mistral-v0.3 (7B) Phi-3.5 (4B)

Phi-4 (15B) Qwen2.5 (3B) Qwen2.5 (7B)

Qwen2.5 (14B) Qwen3 (4B) Qwen3 (8B)

Figure 12: Detailed results on the impact of iteration number of each LLM.

Table 4: Transferability results on unseen datasets.

Model	Size	Method	AdvBench ASR↓	OR-Bench-Toxic ASR↓	XSTest-Toxic ASR↓	OR-Bench ORR↓	XSTest ORR↓	Final F1↑
Llama-3.1	8B	Original	0.58%	3.05%	**0.00%**	48.22%	16.57%	0.7077
		AlphaSteer+	0.58%	3.05%	**0.00%**	48.67%	17.13%	0.7038
		AlphaSteer	0.19%	1.37%	**0.00%**	61.87%	27.07%	0.5921
		Steer	**0.00%**	**0.15%**	**0.00%**	85.60%	49.17%	0.3163
		SCANS	0.19%	1.37%	**0.00%**	72.48%	26.52%	0.4945
		Modified	7.69%	5.80%	4.02%	**46.85%**	**6.63%**	**0.7088**
gemma-2	9B	Original	0.58%	1.98%	**0.00%**	80.52%	28.73%	0.4059
		AlphaSteer+	0.77%	0.92%	0.57%	81.58%	**26.52%**	0.3985
		AlphaSteer	**0.00%**	0.46%	**0.00%**	88.86%	28.73%	0.3103
		Steer	**0.00%**	**0.31%**	**0.00%**	94.16%	56.35%	0.1882
		SCANS	3.08%	5.50%	0.57%	**59.82%**	29.28%	**0.5952**
		Modified	2.88%	1.68%	**0.00%**	82.41%	29.83%	0.3809
Mistral-v0.3	7B	Original	54.42%	48.70%	16.67%	**7.28%**	**3.87%**	0.7920
		AlphaSteer+	52.12%	48.85%	16.67%	7.35%	4.42%	0.7937
		AlphaSteer	45.38%	48.09%	14.37%	7.66%	4.97%	**0.8021**
		Steer	42.69%	40.92%	5.75%	11.75%	11.05%	0.7970
		SCANS	74.42%	41.98%	22.99%	18.57%	7.73%	0.7209
		Modified	**27.31%**	**17.25%**	**2.87%**	37.30%	10.50%	0.7195
Phi-3.5	4B	Original	2.12%	4.89%	1.72%	45.49%	13.26%	0.7234
		AlphaSteer+	**1.15%**	4.43%	1.15%	43.90%	13.81%	**0.7365**
		AlphaSteer	**1.15%**	3.66%	1.15%	48.98%	13.81%	0.7022
		Steer	**1.15%**	3.51%	1.15%	56.41%	22.10%	0.6373
		SCANS	1.54%	**1.37%**	1.15%	79.83%	37.57%	0.3994
		Modified	6.54%	6.41%	**0.00%**	**43.21%**	**12.71%**	0.7306
Phi-4	4B	Original	0.58%	2.14%	**0.00%**	58.83%	17.68%	0.6265
		AlphaSteer+	**0.19%**	**1.53%**	**0.00%**	59.97%	18.23%	0.6182
		AlphaSteer	**0.19%**	**1.53%**	**0.00%**	55.42%	18.23%	0.6551
		Steer	**0.19%**	2.60%	**0.00%**	46.70%	16.57%	0.7201
		SCANS	10.19%	12.98%	0.57%	36.01%	**8.84%**	0.7621
		Modified	0.58%	7.63%	1.15%	**18.73%**	11.60%	**0.8841**
	15B	Original	0.19%	3.97%	**0.00%**	72.71%	17.13%	0.5007
		AlphaSteer+	**0.00%**	3.97%	**0.00%**	72.63%	15.47%	0.5039
		AlphaSteer	**0.00%**	3.36%	**0.00%**	75.97%	16.57%	0.4704
		Steer	**0.00%**	**2.29%**	**0.00%**	84.23%	20.99%	0.3762
		SCANS	1.35%	5.95%	**0.00%**	67.78%	**11.60%**	0.5490
		Modified	0.77%	3.82%	**0.00%**	**53.37%**	**11.60%**	**0.6727**
Qwen2.5	3B	Original	0.19%	2.29%	0.57%	55.42%	19.34%	0.6522
		AlphaSteer+	**0.00%**	2.44%	**0.00%**	53.53%	21.55%	0.6649
		AlphaSteer	0.19%	1.53%	**0.00%**	59.97%	21.55%	0.6144
		Steer	**0.00%**	**0.15%**	**0.00%**	85.97%	37.57%	0.3313
		SCANS	6.35%	16.64%	1.15%	**40.11%**	**13.81%**	**0.7305**
		Modified	0.38%	4.12%	0.57%	54.21%	21.55%	0.6555
	7B	Original	**0.38%**	6.72%	**0.00%**	24.64%	8.84%	0.8569
		AlphaSteer+	0.77%	6.11%	**0.00%**	24.87%	8.84%	0.8563
		AlphaSteer	3.08%	**3.66%**	**0.00%**	34.50%	9.39%	0.8006
		Steer	8.65%	5.50%	**0.00%**	61.03%	29.28%	0.5776
		SCANS	1.54%	6.87%	1.72%	40.56%	11.60%	0.7552
		Modified	3.65%	9.92%	**0.00%**	**19.94%**	**7.73%**	**0.8714**
	14B	Original	**0.00%**	4.58%	**0.00%**	21.15%	8.29%	0.8816
		AlphaSteer+	0.19%	5.34%	**0.00%**	20.77%	8.29%	0.8817
		AlphaSteer	**0.00%**	3.21%	**0.00%**	26.31%	9.39%	0.8551
		Steer	**0.00%**	**0.15%**	**0.00%**	57.01%	16.02%	0.6477
		SCANS	7.88%	16.95%	2.30%	30.40%	19.34%	0.7824
		Modified	0.77%	5.04%	**0.00%**	**17.74%**	**6.08%**	**0.8990**
Qwen3	4B	Original	0.96%	4.73%	0.57%	44.35%	6.63%	0.7402
		AlphaSteer+	7.88%	24.58%	4.02%	**21.08%**	**4.42%**	**0.8307**
		AlphaSteer	33.85%	37.10%	8.62%	22.14%	7.73%	0.7634
		Steer	**0.77%**	**1.83%**	**0.00%**	66.49%	19.89%	0.5583
		SCANS	3.65%	6.11%	0.57%	46.93%	11.60%	0.7107
		Modified	2.31%	4.27%	0.57%	36.69%	7.73%	0.7880
	8B	Original	0.96%	3.21%	1.15%	44.12%	9.39%	0.7419
		AlphaSteer+	7.31%	14.20%	7.47%	62.70%	22.10%	0.5560
		AlphaSteer	**0.58%**	4.89%	1.15%	**34.65%**	**7.18%**	**0.8025**
		Steer	0.96%	4.12%	0.57%	40.03%	9.39%	0.7677
		SCANS	**0.58%**	**2.29%**	**0.00%**	70.96%	20.99%	0.5147
		Modified	0.96%	3.82%	**0.00%**	40.71%	8.29%	0.7651
	14B	Original	0.19%	4.43%	0.57%	40.33%	7.18%	0.7683
		AlphaSteer+	5.96%	22.44%	3.45%	**13.65%**	6.63%	**0.8743**
		AlphaSteer	0.77%	12.21%	0.57%	20.09%	**5.52%**	0.8719
		Steer	**0.00%**	**0.31%**	0.57%	72.33%	20.99%	0.5052
		SCANS	29.04%	26.11%	9.20%	35.94%	25.41%	0.6955
		Modified	4.81%	3.05%	**0.00%**	51.86%	11.05%	0.6801