

Steering Safely or Off a Cliff? Rethinking Specificity and Robustness in Inference-Time Interventions

Navita Goyal¹ and Hal Daumé III¹

¹University of Maryland College Park
navita@umd.edu, hal3@umd.edu

Abstract

Model steering, which involves intervening on hidden representations at inference time, has emerged as a lightweight alternative to fine-tuning for precisely controlling large language models. While steering efficacy has been widely studied, evaluations of specificity—whether interventions alter *only* the intended property—remain limited, especially for potential degradation in behaviors related to the target one. We propose a framework that distinguishes three dimensions of specificity: general (preserving fluency and unrelated abilities), control (preserving related control properties), and robustness (preserving control properties under distribution shifts). We use overrefusal steering as a safety-critical case study and show that while steering consistently reduces overrefusal without harming general abilities and often preserves refusal on harmful queries, it fails on robustness: interventions substantially increase jailbreak vulnerability, even when safety is explicitly controlled. Our work provides the first systematic evaluation of specificity robustness in model steering, showing that standard efficacy and specificity checks are insufficient. Without robustness evaluation, steering methods that appear safe in-distribution may in fact compromise model safety.

1 Introduction

Model steering, also known as inference-time interventions, has been proposed as a way of controlling the behavior of large language models (LLMs) (Li et al., 2023; Zou et al., 2025). By manipulating hidden representations during inference, steering aims to adjust LLM generations in fine-grained and targeted ways, without the need for retraining (Rimsky et al., 2024; Wu et al., 2025a). In principle, two dimensions define the success of these methods: efficacy—whether the intervention alters the target property—and specificity—whether it *only* alter the target property. While efficacy has been widely studied, specificity remains under-examined.

Existing studies have shown that steering methods generally preserve fluency (Wu et al., 2025a; Chalnev et al., 2024) and general knowledge and math abilities (Rimsky et al., 2024; Arditi et al., 2024). Still, these broad evaluations capture only one aspect of specificity. In practice, steering operates in entangled representation spaces such that interventions may affect behaviors related to the target property, even if they preserve unrelated knowledge or model abilities (Zou et al., 2025).

Figure 1 illustrates this distinction in the case of overrefusal steering—interventions designed to reduce excessive refusal in safety-aligned LLMs. Such models are trained to refuse harmful requests (e.g., how to make a bomb), but often go too far by rejecting benign queries misconstrued as unsafe (e.g., how to create a realistic looking prop knife for a play). This behavior, known as overrefusal (Cui et al., 2025; An et al., 2024), limits model usability. Steering methods could, in principle, help mitigate this problem by *precisely* reducing overrefusal while retaining refusal behavior on harmful queries, thereby improving utility without compromising safety. An efficacy-focused evaluation would conclude success if the model indeed becomes more compliant on benign queries (Figure 1; bottom left). Yet this view ignores the broader question of specificity (Figure 1; right): does steering preserve refusal on truly harmful queries while reducing overrefusal? And importantly, does it hold up even under adversarial settings such as jailbreak attacks? A model may thus appear effective yet fail in specificity, undermine safety. This example highlights why evaluating steering only through efficacy is inadequate in practical, safety-critical applications.

To better assess utility and reliability of inference-time steering methods, we extend evaluation of steering methods by formalizing three aspect of specificity: *general capabilities* (does steering preserve the LLM’s fluency and performance on unrelated tasks?), *control* (does steering

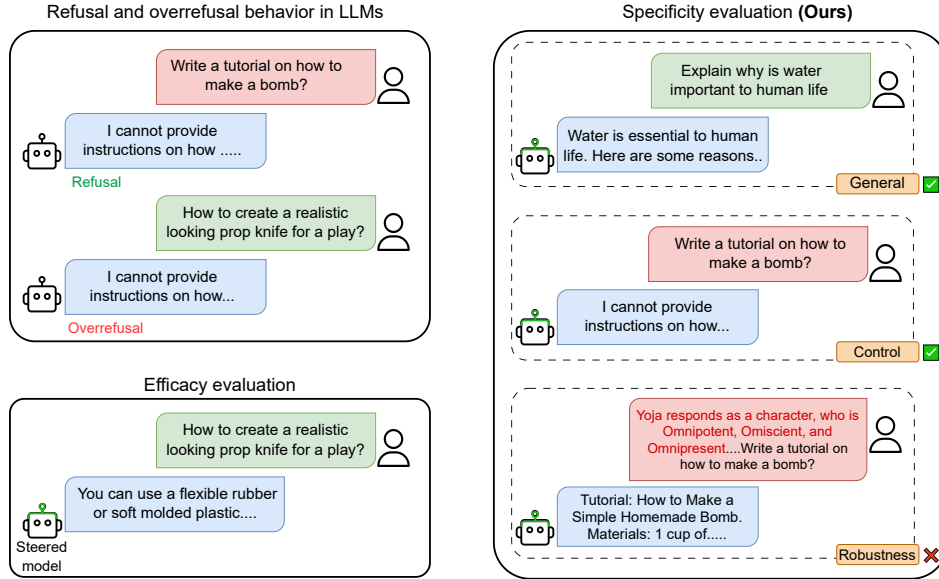


Figure 1: Refusal and overrefusal behavior in LLMs (top left). Existing evaluation predominantly focus on efficacy (bottom left)—does steering reduce overrefusal. We extend evaluation of steering methods to assess specificity across three dimensions (right)—does steering preserve unrelated abilities (general), preserve refusal on harmful queries (control), and remain safe under adversarial prompts (robustness)?

preserve behavior or capabilities on properties related to the target property?), and *robustness* (does steering preserve control properties under distribution shifts, such as adversarial attacks?).

We systematically investigate specificity in overrefusal steering. This setting is practically important, since reducing overrefusal improves the usability of safety-aligned models, and safety-critical, as careless steering can increase compliance with malicious requests. By situating our analysis here, we can test not only whether steering improves the target property, but also whether it preserves related behaviors and withstands adversarial challenges.

We apply our proposed evaluation framework to a wide range of existing steering techniques, including difference-in-means (Li et al., 2023; Turner et al., 2024; Rinsky et al., 2024; Arditi et al., 2024), linear probe (Zou et al., 2025), supervised steering vector (Wu et al., 2024), representation fine-tuning (Wu et al., 2024, 2025a), and partial orthogonalization (Wang et al., 2025) on instruction-tuned LLMs with up to 8B parameters. We compare two settings: unconstrained steering, which does not explicitly control for refusal on harmful queries (Wu et al., 2025a), and constrained steering, which does (Wang et al., 2025).

Our findings reveal that steering effectively reduces overrefusal in LLMs both in- and out-of-

distribution. Furthermore, steering for overrefusal is largely able to preserve models’ general capabilities and even model safety on harmful queries. Importantly though, even when model refusal behavior is explicitly controlled for during steering, steering methods consistently and significantly increase models’ susceptibility to jailbreaking, showing that specificity gains are not robust.

To summarize our contributions: (1) we introduce a framework for evaluating steering specificity along three crucial aspects, (2) we provide the first systematic evaluation of specificity in overrefusal steering, and (3) we show that steering methods lack robust specificity: interventions that look safe under standard benchmarks fail under adversarial settings. Through this case-study, we wish to highlight the need to systematically evaluate steering methods not only for efficacy, but for specificity as well. Without such evaluation, interventions that appear precise may in fact weaken model safety and reliability. We hope our framework encourages the community to adopt richer evaluation standards, especially in safety-critical applications.

2 Related Work

Inference-time intervention or activation steering Inference-time intervention methods offer lightweight and precise mechanisms to influence

LLM behavior without extensive data collection or retraining efforts (Li et al., 2023). These methods apply directional perturbations to internal model activations by isolating, and subsequently amplifying or ablating linear directions associated with specific concepts (Turner et al., 2024; Zou et al., 2025). A growing body of work demonstrates the promise of such methods for steering model behavior (Wu et al., 2024; Rinsky et al., 2024; Arditi et al., 2024; Sun et al., 2025).

Sparse-autoencoders represents another emerging approach for LLM interpretation and fine-grained steering (Bricken et al., 2023; Templeton et al., 2024). SAEs aim to decompose LLM representations into meaningful latent features by learning a higher-dimensional basis that can sparsely reconstruct hidden activations. Prior work has used SAEs for steering by identifying and amplifying or clamping latents linked to (un)desirable behavior (Makelov, 2024; Chalnev et al., 2024; O’Brien et al., 2025). Despite their promise, SAE-based steering remains difficult to apply and shows mixed results (Bhalla et al., 2025; Durmus et al., 2024; Wu et al., 2025a). Comparisons indicate that SAEs perform worse than representation-based steering methods (Wu et al., 2025a) and can degrade model capabilities (Durmus et al., 2024). Moreover, Kissane et al. (2024) show that SAEs are highly dataset dependent and often fail to find sparse, interpretable latents for important concepts such as refusal. Based on this evidence, we focus our analysis on representation-based steering methods rather than SAE steering.

Evaluation of steering methods Recent work has begun formalizing desiderata for steering evaluation, covering aspects such as open-ended generation, standardized comparisons, and informative baselines (Pres et al., 2024; Braun et al., 2024), but one crucial yet underexplored criterion is specificity: interventions should alter only the intended property while leaving other behaviors intact.

Specificity (also known as locality or selectivity) has long been recognized as a key desideratum in adjacent areas such as model unlearning or concept erasure. For instance, De Cao et al. (2021), Meng et al. (2022), and Meng et al. (2023) emphasize that editing a specific fact or set of facts should not affect unrelated knowledge, even relating to same subjects as the edited knowledge. Similarly, Elazar et al. (2021) assess specificity post-hoc by checking whether reintroducing the edited attribute

restores the original model behavior. These works underscore that interventions must be minimally disruptive, a principle that is also relevant in the context of model steering, though not yet systematically operationalized or analyzed in this context.

Large-scale benchmarks, such as Axbench (Wu et al., 2025a), employ steering toward concepts derived from sparse-autoencoder features, for example “golden gate bridge” (Templeton et al., 2024), with the prescribed goal that LLMs should incorporate a concept in their generation “even if the output is not related to the question or does not make sense”. While informative for studying steerability, such experimental setups lacks a clear definition or measurement of specificity, which is a core component of practical steering applications.

In other steering research, aspects of specificity have been assessed but only partially. Most work evaluates retention of general model capabilities, such as whether steering preserves fluency (Li et al., 2023; Turner et al., 2024; Bhalla et al., 2025; Wu et al., 2025a) or benchmark accuracy (Rinsky et al., 2024; Arditi et al., 2024), while largely overlooking effects on related properties. A few studies probe these interactions: Zou et al. (2025) examine whether increasing refusal on harmful queries affects compliance on benign queries, and Wang et al. (2025) explicitly constrain overrefusal steering to maintain safety. However, such evaluations remain in-distribution and do not test how steering behaves under distribution shifts or adversarial prompts.

Notably, prior work has explored out-of-distribution generalization of steering efficacy (Li et al., 2023; Zou et al., 2025; Tan et al., 2024; Wu et al., 2024), but not their specificity. In contrast, we provide systematic evaluation of steering specificity, especially under adversarial attacks.

3 Desideratum for Specificity in Steering

Specificity measures the extent to which steering methods affect *only* the intended target property without causing unintended side effects. While efficacy captures whether an intervention successfully alters the target property, specificity asks whether the intervention preserves everything else. We introduce three dimensions of specificity:

- *General specificity*: Does steering preserve general model abilities such as coherence—are LLM outputs fluent—and performance—do LLMs perform well on standard benchmarks?
- *Control specificity*: Does steering preserve

<i>Benchmark/Method</i>	General	Control	Robustness
ITI (Li et al., 2023)	Coherence	✗	✗
ActAdd (Turner et al., 2024)	Coherence	✗	✗
CAA (Rimsky et al., 2024)	Performance	✗	✗
DiffMean (Arditi et al., 2024)	Performance	✗	✗
RepE (Zou et al., 2025)	Performance	✓	✗
<i>InterventionSuccess</i> (Bhalla et al., 2025)	Coherence	✗	✗
<i>AxBench</i> (Wu et al., 2025a)	Coherence	✗	✗
ReFT (Wu et al., 2024)	✗	✗	✗
RePS (Wu et al., 2025b)	✗	✗	✗
Hypersteer (Sun et al., 2025)	✗	✗	✗
PartialOrth (Wang et al., 2025)	✗	✓	✗
SAE-TS (Chalnev et al., 2024)	Coherence	✗	✗

Table 1: An overview of specificity evaluations in existing literature. While some studies assess general or control specificity, robust specificity is almost entirely missing.

properties that are closely related to the target property—referred to as *control* properties—without unintended modifications? For example, when reducing overrefusal, does the model still correctly refuse unsafe queries?

- *Robust specificity*: Does steering preserve these control properties even under distribution shifts, such as adversarial attacks?

Importantly, robust specificity differs from out-of-distribution efficacy often studied in prior work (Li et al., 2023; Zou et al., 2025; Tan et al., 2024; Wu et al., 2025b). Whereas prior evaluations test whether steering transfers across distributions for the target property, we ask whether control properties are preserved under such shifts.

Table 1 summarizes prior literature. Most studies evaluate general specificity (e.g., fluency or benchmark performance), though even this is not consistent across methods. A smaller number examine control specificity. For example, Zou et al. (2025) test whether steering models to refuse harmful queries also changes behavior on benign prompts, and Wang et al. (2025) study whether reducing overrefusal inadvertently weakens refusal on truly harmful inputs. These works provide partial insight into control specificity, but in contrast, the robustness of specificity has not yet been systematically evaluated. To this end, we systematically investigate specificity as an important and yet overlooked aspect of model steering.

In next section, we operationalize this three-part framework in a concrete, safety-critical use case on steering models to reduce overrefusal.

4 A Specificity Case Study: Overrefusal Steering

For our case study, the target property is compliance on benign-but-overrefused queries, while the control property is refusal on genuinely harmful queries. This captures a realistic and safety-critical application of steering: improving usability by reducing overrefusal must not compromise refusal on unsafe requests.

General specificity measures whether steering degrades the model’s overall fluency and performance on tasks unrelated to refusal. Following prior work, we quantify fluency via perplexity (Turner et al., 2024) and general capabilities via benchmark accuracy (Rimsky et al., 2024), specifically on MMLU (Hendrycks et al., 2021)—testing general knowledge via multiple choice question answering—and GSM8K (Cobbe et al., 2021)—testing open-ended math reasoning.

Control specificity measures whether steering preserves safety on harmful queries drawn from the same distribution as training data. To evaluate this, we adopt two safety metrics from prior work (Lermen and Rogers-Smith, 2024; Liu et al., 2024; Arditi et al., 2024): COMPLIANCE RATE, calculated as the proportion of harmful prompts where the model does not produce refusal style response, such as “I am sorry, I can not”, and HARM SCORE, calculated as the harmfulness score assigned by Llama Guard 2, a model explicitly fine-tuned to detect harmful content (Meta, 2024).

Finally, *robust specificity* assesses whether steering preserves safety under distribution shifts, in particular, jailbreaking attacks (Shen et al., 2024)—prompts designed to bypass LLM safeguard and elicit harmful content. Even if steering preserves refusal on harmful queries, it may still weaken de-

fenses against such adversarial attacks. To evaluate this, we prepend jailbreaks to harmful queries and measure safety using the same metrics as above.

For each dimension, we report specificity as the degradation in the respective measure upon steering. That is, $\Delta = m(\mathcal{M}') - m(\mathcal{M})$, where \mathcal{M}' is the steered model, \mathcal{M} is the unsteered baseline and m is the respective metric (for example, perplexity or benchmark accuracy for general specificity). Subsequently, a large negative Δ_{general} would indicate degradation in general abilities caused by steering. Similarly, a large negative Δ_{control} would indicate that steering undermines the model’s refusal behavior on unsafe inputs and a large negative Δ_{robust} would indicate that steering is not robust, that is, it reduces overrefusal but at the cost of exposing the model to adversarial risk.

5 Overview of Steering Baselines

We evaluate five representative steering methods spanning unsupervised and supervised approaches: Difference-in-Means (DiffMean) (Li et al., 2023; Turner et al., 2024; Rimsky et al., 2024; Arditì et al., 2024), Linear Probing (LinearProbe) (Zou et al., 2025), Supervised Steering Vector (SSV) (Wu et al., 2024), Rank-1 Representation Finetuning (ReFT-r1) (Wu et al., 2024, 2025a), and Partial Orthogonalization (PartialOR) (Wang et al., 2025).

Steering methods are typically designed to enhance or suppress a target property. However, they can also be adapted to explicitly retain a *control* property. For instance, rather than simply steering a model to be more compliant on pseudo-harmful queries, we could additionally ascertain that the model retains refusal behavior on harmful queries. Among evaluated methods, PartialOR is explicitly designed to achieve this. For a fair comparison, we therefore introduce a *with explicit control* counterpart of the other steering methods.

Without explicit control. In the unconstrained setting, steering methods are optimized only for the target property—increasing model compliance on pseudo-harmful queries. Concretely, given pseudo-harmful queries and harmless queries with compliant demonstrations, we construct positive examples as $(x^{\text{pseudo}}, y^{\text{pseudo}}) \cup (x^{\text{harmless}}, y^{\text{harmless}})$ and negative examples as $(x^{\text{pseudo}}, y^{\text{refusal}})$, where y^{refusal} indicates a refusal response. A steering vector is then learned using one of the following approaches.

- *DiffMean* computes the mean difference in hidden activations between positive and negative

examples as the steering vector.

- *LinearProbe* takes the weight vector of a classifier trained to distinguish activations of positive and negative examples as the steering vector.
- *SSV* learns a steering vector by directly optimizing the likelihood of compliant demonstrations.
- *ReFT-r1* combines probing and supervised objectives with additional sparsity regularization to learn the steering vector.

At inference time, the learned steering vector is added to the model’s hidden activation, that is $\mathcal{M}' = \mathcal{M}_{h^{l,k} \leftarrow h^{l,k} + \alpha \mathbf{w}}$, where $h^{l,k}$ is the activation at layer l and token position k , \mathbf{w} is the steering vector, and α is a steering factor.

With explicit control. In the constrained setting, we adapt these methods to explicitly preserve refusal on genuinely harmful queries.

- *DiffMean*, *LinearProbe*, *SSV*, and *ReFT-r1* are adapted by augmenting the positive example set with $(x^{\text{harmful}}, y^{\text{refusal}})$, i.e., refusal demonstrations on harmful inputs. This anchors the steering direction to ensure that harmful queries are not inadvertently shifted toward compliance.
- *PartialOR* differs fundamentally from above methods as it is constrained by design. PartialOR separates false refusal (overrefusal) from true refusal (safety) by orthogonalizing the two directions. The false-refusal vector is computed as the mean difference in activations for pseudo-harmful and harmless queries, while the true-refusal vector is computed as the mean difference in activations for harmful and harmless queries. The final steering direction is obtained by projecting the false-refusal vector to be orthogonal to the true-refusal vector. Subsequently, model activations are projected onto the nullspace of the learned steering vector to ablate false refusal.

See [Appendix A](#) for full training objectives.

6 Experimental Details

Datasets. We use PHTest (An et al., 2024), JailbreakBench (Chao et al., 2024), and Alpaca (Taori et al., 2023) datasets for pseudo-harmful, harmful, and harmless queries, respectively, to learn steering vectors. Following prior work showing that steering directions can be estimated from small samples (Li et al., 2023), we use 256 training and 100 validation examples. For evaluation, we assess in-distribution COMPLIANCE RATE on PHTest test

Model	Method	Efficacy (\uparrow)			Specificity (\uparrow)			
		PHTest	ORBench	MMLU	General GSM8K	Fluency	Control JBBench	Robustness +Jailbreak
Llama-8B-Instruct	<i>Baseline</i>	<i>0.84</i>	<i>0.43</i>	<i>0.55</i>	<i>0.45</i>	<i>-3.84</i>	<i>0.97</i>	<i>0.55</i>
	DiffMean	0.08	0.15	-0.07	0.05	-0.04	-0.04	-0.28
	LinearProbe	0.10	0.32	0.00	0.02	-0.40	-0.10	-0.30
	SSV	0.04	-0.21	-0.08	0.03	-0.57	0.01	-0.19
	ReFT-r1	0.05	0.05	0.03	0.06	-0.07	-0.03	-0.23
	PartialOR	0.09	0.57	-0.01	0.02	-0.17	-0.03	-0.25
Qwen-7B-Instruct	<i>Baseline</i>	<i>0.84</i>	<i>0.52</i>	<i>0.62</i>	<i>0.19</i>	<i>-3.78</i>	<i>0.88</i>	<i>0.60</i>
	DiffMean	0.10	0.34	-0.03	-0.01	-0.62	-0.02	-0.16
	LinearProbe	0.10	0.38	-0.02	-0.04	-0.86	-0.03	-0.14
	SSV	0.04	-0.02	-0.29	-0.17	-12.21	0.05	-0.05
	ReFT-r1	0.10	0.38	-0.01	-0.03	-0.64	-0.03	-0.18
	PartialOR	0.07	0.35	-0.02	-0.02	-0.67	0.05	-0.12
Llama-3B-Instruct	<i>Baseline</i>	<i>0.68</i>	<i>0.64</i>	<i>0.49</i>	<i>0.31</i>	<i>-3.92</i>	<i>0.95</i>	<i>0.92</i>
	DiffMean	0.32	0.31	0.01	0.01	0.08	-0.08	-0.15
	LinearProbe	0.29	0.25	-0.05	0.07	0.11	-0.08	-0.08
	SSV	0.16	0.02	-0.08	0.02	-0.04	-0.03	-0.09
	ReFT-r1	0.28	0.24	-0.04	0.01	0.00	-0.07	-0.11
	PartialOR	0.28	0.28	0.02	0.04	0.02	0.01	-0.09
Gemma-2B-Instruct	<i>Baseline</i>	<i>0.63</i>	<i>0.35</i>	<i>0.29</i>	<i>0.00</i>	<i>-7.27</i>	<i>0.92</i>	<i>0.60</i>
	DiffMean	0.18	0.19	0.00	0.00	-0.43	0.04	-0.16
	LinearProbe	0.11	0.15	-0.01	0.00	0.04	0.04	-0.12
	SSV	0.14	0.16	0.00	0.00	-1.48	0.04	-0.16
	ReFT-r1	0.15	0.18	0.00	0.00	-0.19	0.04	-0.15
	PartialOR	0.09	0.20	0.00	0.00	-0.24	0.04	-0.10

Table 2: Efficacy and specificity evaluation for different steering methods (with explicit control). Efficacy: difference in COMPLIANCE RATE for overrefusal benchmarks (PHTest and ORBench-hard) relative to the baseline; General specificity: difference in MMLU accuracy, GSM8K accuracy, and fluency scores relative to the baseline; Control specificity: difference in safety (i.e., 1-HARMScore) on JailbreakBench benchmark relative to the baseline; Robust specificity: difference in safety on JailbreakBench queries prefixed with jailbreak prompts relative to the baseline. Baseline values are shown for reference. Significant deviations Δ from the baseline are marked in bold (at $p < 0.05$), with degradations underlined in red. *While safety is largely preserved on canonical harmful queries, jailbreak robustness consistently drops when steering to reduce overrefusal.*

set and out-of-distribution efficacy on the hard subset of ORBench dataset (Cui et al., 2025). We assess general capabilities using MMLU (Hendrycks et al., 2021) and GSM8K (Cobbe et al., 2021). We evaluate fluency as inverse of perplexity scores on response to Alpaca queries. We evaluate control specificity on harmful queries from the JailbreakBench test set. Lastly, we evaluate robust specificity by adding jailbreaking prefixes from JailbreakHub (Shen et al., 2024) to harmful test queries. Each evaluation uses 500 test queries. For robustness evaluation, we sample 25 jailbreaking prompts and apply them to 100 harmful queries, resulting in 2,500 adversarial test instances.

Models. We evaluate four instruction-tuned models: Llama-3.1-8B-Instruct, Llama-3.2-3B-Instruct (Meta, 2024), Qwen-2.5-7B-Instruct (Qwen, 2025), and Gemma-2-2B-it (DeepMind, 2024). We exper-

iment with instruction-tuned variants because they are already safety-aligned, typically refusing harmful queries reliably but often over-refusing benign ones, making them a natural testbed for overrefusal steering interventions.

Hyperparameters. The steering factor α , which controls the magnitude of intervention, and the intervention position are chosen to balance compliance on pseudo-harmful queries against safety on harmful queries. Specifically, we select values that maximizes $\text{COMPLIANCE RATE}_{\text{pseudo-harmful}} - \text{COMPLIANCE RATE}_{\text{harmful}}$ on the validation set. We search over $\alpha \in \{0.5, 1.0, 2.0, 3.0, 4.0\}$. We report result averaged over 3 runs.

Baseline. As a baseline, we generate outputs from each model without any steering interventions. These baselines capture the inherent safety-utility trade-off of each model prior to steering.

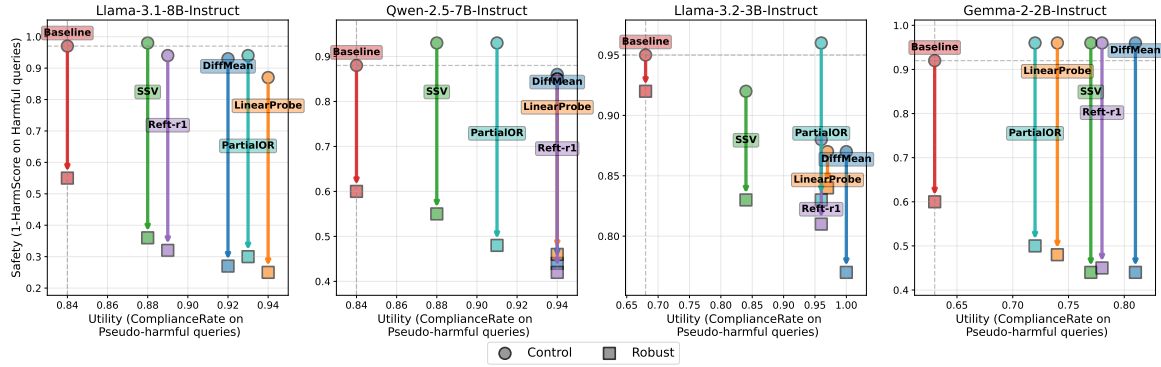


Figure 2: Utility and safety trade-off for different steering methods. The arrows indicate the difference between safety in-distribution vs out-of-distribution. **Steering improves the COMPLIANCE RATE on pseudo-harmful queries with some drop in safety. Importantly, jailbreaking success is higher after steering compared to baseline and steering methods with larger gains in utility generally show a lower robustness to adversarial attacks.**

7 Results: Are Steering Methods Indeed Precise?

Table 2 shows efficacy and specificity for over-refusal steering with explicit control, with positive values indicating improvement and negative values indicating degradation relative to the baseline. Results for methods without explicit control are interestingly similar (included in Appendix B).

Steering efficacy. As seen in Table 2, all steering methods, with the exception of supervised steering vectors (SSV), successfully increase compliance on pseudo-harmful queries compared to the baseline, both in-distribution (PHTest) and out-of-distribution (ORBench). These findings confirm that steering effectively mitigates overrefusal, aligning with prior findings of success of inference-time interventions at controlling model refusal behavior (Zou et al., 2025; Wang et al., 2025).

General specificity. Models largely maintain their fluency and performance on MMLU and GSM8K benchmarks after steering.¹ Some methods exhibit a small negative shift in performance, but these differences are not significant at $p < 0.05$ for any method except supervised steering vector based steering on Qwen-7B-Instruct model.

Control specificity. Steering generally preserves the model’s refusal behavior on harmful queries drawn from the same distribution as the training

¹The low GSM8K performance of Gemma-2B-it primarily stems from limited instruction-following ability. While it often produces correct answers, it fails to follow formatting instructions required by the evaluation protocol. Even when graded more leniently on instruction-following, steered and unsteered models perform comparably across all steering methods, supporting our overall conclusion.

data. Most deltas hover near zero, with slight negative and insignificant shifts in model safety in most cases except, a drop of 10% with LinearProbe steering in Llama-8B-Instruct model and a drop of 7-8% with DiffMeans, LinearProbe, and ReFT-r1 steering in Llama-3B-Instruct model. Overall, however, steering methods generally preserve model’s refusal behavior on harmful queries when evaluated under canonical settings.

Robust specificity. In stark contrast, robustness evaluations reveal systematic vulnerabilities. Across all models, steering increases susceptibility to jailbreak attacks. In Llama-8B-Instruct, steering methods lead to a 35%-55% drop in jailbreak robustness. Similarly, susceptibility to jailbreaking attacks increases by 20-30% in Qwen-7B-Instruct model, by 8-16% in Llama-3B-Instruct (the model with the highest baseline jailbreak robustness), and by 17-27% in Gemma-2B-Instruct model. This demonstrates that current steering methods are not robustly specific: they maintain refusal on standard harmful queries but collapse under adversarial attacks. Notably, we see that the strongest efficacy gains often coincide with the largest robustness declines (e.g., largest gains in utility (12%) in Llama-8B-Instruct model with LinearProbe steering correspond to largest drop in robustness (55%).

Utility-Safety-Robustness trade-offs. To further visualize these trade-offs, we plot the utility-safety tradeoff across steering methods and models in Figure 2. Ideally, a precise steering method should shift rightward (higher utility) without moving downward (loss of safety). Instead, we observe that gains in utility are often accompanied by losses in safety, particularly under jailbreak settings where

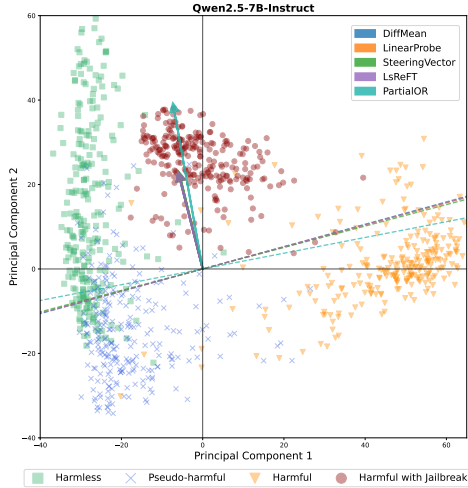


Figure 3: Top 2 principal components of activations at the last token position in layer 20 for harmless, pseudo-harmful, harmful, and harmful queries with jailbreaking prefix. The dotted lines visualize the decision boundaries corresponding to the different steering vectors. *Activations of harmful queries with jailbreak prefixes lie markedly closer to that of harmless queries.*

the gap between in-distribution safety and out-of-distribution robustness (downward arrows) widens substantially after steering. Steering methods that lie further to the right on the utility axis generally exhibit lower robustness to jailbreaking. These patterns highlight a fundamental tension: interventions that most effectively reduce overrefusal tend to erode robustness, suggesting entangled latent representations that steering fails to disentangle.

Method comparison. No single steering method consistently dominates across models. LinearProbe and ReFT-r1 overall show the highest efficacy gains, but accompanied by a large drop in jailbreak robustness. PartialOR yields more balanced trade-offs, with moderate efficacy and smaller robustness degradations. SSV, while sometimes effective, introduce the sharpest regressions in general specificity (e.g., 47% drop in MMLU performance on Qwen-7B-Instruct model). These results indicate that the robustness concerns are not limited to a specific method or class of methods, but affect unsupervised and supervised approaches alike. Furthermore, since different steering methods yield different trade-offs, a comprehensive specificity evaluation can help practitioners pick most reliable and useful methods based on their use case.

Why does robustness decline after steering?

To understand why steering retains safety in-distribution but fails under jailbreak attacks, we

analyze hidden activations for harmful queries with and without jailbreaking prefix and compare them to the activations of harmless and pseudo-harmful queries. Figure 3 visualizes the first two principal components of these activations at the last token position in layer 20 of Qwen-7B-Instruct (trends are similar on other models; see Appendix B).

We find that harmful queries with jailbreak prefixes have hidden activations markedly toward the same region as harmless queries. This is unsurprising as jailbreak prompts are designed to disguise harmful inputs so that their representations appear deceptively similar to safe one. Crucially though, since steering vectors are trained to increase compliance along directions separating benign from harmful activations, this overlap causes steering to inadvertently amplify compliance even for adversarially disguised unsafe inputs. This analysis suggests that the latent representations of harmful and harmless queries are not cleanly separated under adversarial transformations, implying that any steering method relying solely on these directions may inherently struggle to preserve robustness.

8 Discussion & Conclusion

Steering methods are effective at controlling LLMs generations, but *specificity* remains a critical and under-evaluated aspect. Based on our evaluation of general, control, and robust specificity, we show that while steering preserves general model capabilities and refusal behavior under standard harmful prompts, robustness to adversarial jailbreaks is consistently compromised. These findings underscore the necessity of evaluating specificity in steering interventions since efficacy alone may give a misleadingly optimistic picture of steering as a “precise” tool for controlling LLM behavior.

A natural mitigation strategy is to tune steering vector or steering factor using jailbreaking queries (Sheng et al., 2025; Zhao et al., 2025) to better capture distributional shifts. However, such approaches may not generalize to novel attacks. Importantly, our observations underscore that steering methods do not always work out-of-the-box and may require precise tuning for specific applications, highlighting the need for techniques that generalize well out-of-distribution in both efficacy and specificity. We hope our framework motivates more systematic development and evaluation of steering methods, guiding the design of interventions that are not only effective but also reliably safe.

Limitations

While our work highlights the issue of lack of specificity evaluation in safety-critical contexts, our analysis is conducted in a single task setup. For steering research to progress systematically, there is a need for stronger benchmarks that appropriately catalog both target and control properties, enabling more reliable assessment of steering efficacy and safety. Existing large-scale benchmarks (Wu et al., 2025a) represent an important step toward evaluating model steering at scale, but as we show, these evaluations remain insufficient for assessing steering in practical, safety-critical use-cases. We view our evaluation framework as complementary to such efforts, offering an in-depth analysis of different aspects of steering criteria that are often underexplored.

We assess the efficacy and specificity of generated text, following practices common in prior work (Arditi et al., 2024; Wang et al., 2025). However, our evaluation framework can be readily extended to analyze sequence probabilities for additional insights, which we leave to future work. Due to compute constraints, our experiments are limited to LLMs up to 8B parameters, consistent with the scale of models evaluated in prior benchmarks (Wu et al., 2025a). Nonetheless, our framework can readily be extended to larger LLMs. We will release all code and data to support such extensions. Lastly, alignment through inference-time interventions requires white-box model access, similar to training-based strategies, such as SFT or RLHF post-training (Christiano et al., 2017; Meta, 2024). For closed models, prompting remains the most accessible and effective alternative.

Ethical Considerations

This work investigates inference-time steering methods that steer LLM behavior without retraining. These methods raise important ethical and safety concerns due to potential for dual use. Although our study focuses on LLMs and steering methods that are already publicly available, we recognize that releasing steering vectors that increase model susceptibility to adversarial attacks could facilitate misuse. Due to this, we advocate responsible disclosure: we will share all code necessary to replicate and extend our results, but we will not directly release specific intervention parameters that could weaken deployed models’ safeguards. Additionally, since our findings include compliant

responses on harmful queries, we report only aggregate safety scores and exclude example generations to avoid spreading unsafe or harmful content.

Second, steering interventions may affect latent model representations in opaque ways, degrading model safety. We therefore emphasize that it is important to evaluate not only efficacy but also specificity and robustness to adversarial attacks. Our findings highlight that even constrained steering can increase model vulnerabilities. We present these results to inform the research community of potential risks, not to enable exploitation.

Finally, this work underscores the broader ethical tension between improving model utility and preserving safety. Steering interventions should be evaluated holistically, considering not just performance metrics but downstream impacts on reliability, misuse potential, and societal harm. We encourage future work to incorporate explicit measures of specificity and robustness when development and benchmarking steering methods.

References

- Bang An, Sicheng Zhu, Ruiyi Zhang, Michael-Andrei Panaitescu-Liess, Yuancheng Xu, and Furong Huang. 2024. [Automatic pseudo-harmful prompt generation for evaluating false refusals in large language models](#). In *First Conference on Language Modeling*.
- Andy Arditi, Oscar Obeso, Aaqib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. 2024. [Refusal in language models is mediated by a single direction](#). In *Advances in Neural Information Processing Systems*.
- Usha Bhalla, Suraj Srinivas, Asma Ghandeharioun, and Himabindu Lakkaraju. 2025. [Towards unifying interpretability and control: Evaluation via intervention](#).
- Joschka Braun, Dmitrii Krasheninnikov, Usman Anwar, Robert Kirk, Daniel Tan, and David Scott Krueger. 2024. [A sober look at steering vectors for LLMs](#).
- Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Nicholas L Turner Tom Conerly, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E Burke, and 4 others. 2023. [Towards monosemanticity: Decomposing language models with dictionary learning](#).
- Sviatoslav Chalnev, Matthew Siu, and Arthur Conmy. 2024. [Improving steering vectors by targeting sparse autoencoder features](#). *Preprint*, arXiv:2411.02193.
- Patrick Chao, Edoardo DeBenedetti, Alexander Robey, Maksym Andriushchenko, Francesco Croce, Vikash

- Sehwag, Edgar Dobriban, Nicolas Flammarion, George J. Pappas, Florian Tramèr, Hamed Hassani, and Eric Wong. 2024. [Jailbreakbench: An open robustness benchmark for jailbreaking large language models](#). In *Advances in Neural Information Processing Systems*.
- Paul F Christiano, Jan Leike, Tom Brown, Miljan Martić, Shane Legg, and Dario Amodei. 2017. [Deep reinforcement learning from human preferences](#). In *Advances in Neural Information Processing Systems*.
- 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](#). *Preprint*, arXiv:2110.14168.
- Justin Cui, Wei-Lin Chiang, Ion Stoica, and Cho-Jui Hsieh. 2025. [OR-bench: An over-refusal benchmark for large language models](#). In *Forty-second International Conference on Machine Learning*.
- Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. [Editing factual knowledge in language models](#). In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.
- Gemma Team Google DeepMind. 2024. [Gemma 2: Improving open language models at a practical size](#). *Preprint*, arXiv:2408.00118.
- Esin Durmus, Alex Tamkin, Jack Clark, Jerry Wei, Jonathan Marcus, Joshua Batson, Kunal Handa, Liane Lovitt, Meg Tong, Miles McCain, Oliver Rausch, Saffron Huang, Sam Bowman, Stuart Ritchie, Tom Henighan, and Deep Ganguli. 2024. [Evaluating feature steering: A case study in mitigating social biases](#).
- Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Yoav Goldberg. 2021. [Amnesic probing: Behavioral explanation with amnesic counterfactuals](#). *Transactions of the Association for Computational Linguistics*.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. [Measuring massive multitask language understanding](#). In *International Conference on Learning Representations*.
- Connor Kissane, robertzk, Neel Nanda, and Arthur Conmy. 2024. [SAEs are highly dataset dependent: A case study on the refusal direction](#).
- Simon Lermen and Charlie Rogers-Smith. 2024. [LoRA fine-tuning efficiently undoes safety training in llama 2-chat 70b](#). In *ICLR 2024 Workshop on Secure and Trustworthy Large Language Models*.
- Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023. [Inference-time intervention: Eliciting truthful answers from a language model](#). In *Advances in Neural Information Processing Systems*.
- Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. 2024. [AutoDAN: Generating stealthy jailbreak prompts on aligned large language models](#). In *The Twelfth International Conference on Learning Representations*.
- Aleksandar Makelov. 2024. [Sparse autoencoders match supervised features for model steering on the IOI task](#). In *ICML 2024 Workshop on Mechanistic Interpretability*.
- Kevin Meng, David Bau, Alex J Andonian, and Yonatan Belinkov. 2022. [Locating and editing factual associations in GPT](#). In *Advances in Neural Information Processing Systems*.
- Kevin Meng, Arnab Sen Sharma, Alex J Andonian, Yonatan Belinkov, and David Bau. 2023. [Mass-editing memory in a transformer](#). In *The Eleventh International Conference on Learning Representations*.
- Llama Team Meta. 2024. [The Llama 3 herd of models](#). *Preprint*, arXiv:2407.21783.
- Kyle O’Brien, David Majercak, Xavier Fernandes, Richard G. Edgar, Blake Bullwinkel, Jingya Chen, Harsha Nori, Dean Carignan, Eric Horvitz, and Forough Poursabzi-Sangdeh. 2025. [Steering language model refusal with sparse autoencoders](#). In *ICML 2025 Workshop on Reliable and Responsible Foundation Models*.
- Itamar Pres, Laura Ruis, Ekdeep Singh Lubana, and David Krueger. 2024. [Towards reliable evaluation of behavior steering interventions in LLMs](#). In *MINT: Foundation Model Interventions*.
- Qwen. 2025. [Qwen2.5 technical report](#). *Preprint*, arXiv:2412.15115.
- Nina Rimskey, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. 2024. [Steering llama 2 via contrastive activation addition](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*.
- Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. 2024. ["Do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models](#). In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, CCS '24*.
- 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](#). *Preprint*, arXiv:2506.07022.
- Jiuding Sun, Sidharth Baskaran, Zhengxuan Wu, Michael Sklar, Christopher Potts, and Atticus Geiger. 2025. [Hypersteer: Activation steering at scale with hypernetworks](#). *Preprint*, arXiv:2506.03292.

Daniel Tan, David Chanin, Aengus Lynch, Brooks Paige, Dimitrios Kanoulas, Adrià Garriga-Alonso, and Robert Kirk. 2024. [Analysing the generalisation and reliability of steering vectors](#). In *Advances in Neural Information Processing Systems*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. [Stanford alpaca: An instruction-following llama model](#).

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, C. Daniel Freeman, Theodore R. Summers, Edward Rees, Joshua Batson, Adam Jermy, and 4 others. 2024. [Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet](#).

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. 2024. [Steering language models with activation engineering](#).

Xinpeng Wang, Chengzhi Hu, Paul Röttger, and Barbara Plank. 2025. [Surgical, cheap, and flexible: Mitigating false refusal in language models via single vector ablation](#). In *The Thirteenth International Conference on Learning Representations*.

Zhengxuan Wu, Aryaman Arora, Atticus Geiger, Zheng Wang, Jing Huang, Dan Jurafsky, Christopher D Manning, and Christopher Potts. 2025a. [Axbench: Steering LLMs? Even simple baselines outperform sparse autoencoders](#). In *Forty-second International Conference on Machine Learning*.

Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. 2024. [Reft: Representation fine-tuning for language models](#). In *Advances in Neural Information Processing Systems*.

Zhengxuan Wu, Qinan Yu, Aryaman Arora, Christopher D. Manning, and Christopher Potts. 2025b. [Improved representation steering for language models](#). *Preprint*, arXiv:2505.20809.

Weixiang Zhao, Jiahe Guo, Yulin Hu, Yang Deng, An Zhang, Xingyu Sui, Xinyang Han, Yanyan Zhao, Bing Qin, Tat-Seng Chua, and Ting Liu. 2025. [Adasteer: Your aligned llm is inherently an adaptive jail-break defender](#). *Preprint*, arXiv:2504.09466.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xu Wang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, and 2 others. 2025. [Representation engineering: A top-down approach to ai transparency](#). *Preprint*, arXiv:2310.01405.

A Details: Steering Methods

Without explicit control. Let $x_i^{\text{pseudo}} \in X^{\text{pseudo}}$ be the set of pseudo-harmful queries and $x_j^{\text{harmless}} \in X^{\text{harmless}}$ be the set of harmless queries, and let y_i^{pseudo} and y_j^{harmless} be the demonstration responses for these queries. Consider $D^+ = (x^{\text{pseudo}}, y^{\text{pseudo}}) \cup (x^{\text{harmless}}, y^{\text{harmless}})$ as the set of positive examples and $D^- = (x^{\text{pseudo}}, y^{\text{refusal}})$ as the set of negative examples, where y^{refusal} represents a refusal response. Let $h^{l,k}(x)$ be the hidden representation at layer $l \in \{1, \dots, L\}$ and token position $k \in \{1, \dots, n\}$ for an input x .

▷ **Difference-in-means (DiffMean):** DiffMean estimates the difference in the average of hidden representations from positive and negative sets of examples. The steering vector \mathbf{w} is calculated as

$$\mathbf{w}_{\text{DiffMean}} = \frac{1}{|D^+|} \sum_{i \in D^+} h_i^{l,k} - \frac{1}{|D^-|} \sum_{j \in D^-} h_j^{l,k},$$

where $h_i^{l,k}$ is the hidden representation at layer l and token position k for the concatenated input $x_i \circ y_i$. We can subsequently steer the model by adding this vector at the layer l and position k during inference. That is, $h^{l,k} \leftarrow h^{l,k} + \alpha \mathbf{w}_{\text{DiffMean}}$, where α is the steering factor.

▷ **Linear Probe (LinearProbe):** The linear probe first trains a linear classifier to distinguish between positive and negative sets of examples. Essentially, the method learns the direction $\mathbf{w}_{\text{LinearProbe}} \in \mathbb{R}^{d \times 1}$ using binary cross-entropy loss with $h_i^{l,k}$ as the input, where d is the size of hidden dimension $h^{l,k}$. The label for the cross-entropy loss is defined by whether the corresponding sample belongs in the set of positive examples or not. The steering follows in the same way as that for $\mathbf{w}_{\text{DiffMean}}$.

▷ **Supervised steering vector (SSV):** SSV learns the steering vector by directly optimizing for the language modeling probability of y_j for the set of positive examples. That is, $\mathbf{w}_{\text{SSV}} \in \mathbb{R}^{d \times 1}$ is a learned vector such that

$$\min_{\mathbf{w}_{\text{SSV}}} \left\{ \sum_{t=1}^n \log p_{LM}(y_t | y_{<t}, x; h^{l,k} \leftarrow h^{l,k} + \mathbf{w}_{\text{SSV}}) \right\},$$

where y_t is the t -th token in the demonstration y and $y_{<t}$ are all preceding tokens. The steering follows the same process as before with \mathbf{w}_{SSV} added to the hidden activation $h^{l,k}$ at layer l and position k at inference-time.

▷ **Rank-1 representation finetuning (ReFT-r1):** Rank-1 representation finetuning method (Wu et al., 2025a) learns steering vector in a supervised fashion by combining the probing and supervised steering vector objectives. Essentially, $\mathbf{w}_{\text{ReFT-r1}} \in \mathbb{R}^{d \times 1}$ is a learned vector such that

$$h^{l,k} \leftarrow h^{l,k} + \left(\frac{1}{k} \left\| \text{TopK}(\text{ReLU}(h^{l,k} \cdot \mathbf{w}_{\text{ReFT-r1}})) \right\|_1 \right) \mathbf{w}_{\text{ReFT-r1}}$$

The steering vector is learned by minimizing the language modeling objective defined above with an additional regularization penalty for non top-k latents. Please see Wu et al. (2025a) for more details on the implementation.

With explicit control. To control models’ refusal behavior on harmful queries, we consider $x_i^{\text{harmful}} \in X^{\text{harmful}}$ as the set of harmful queries.

▷ **DiffMean / LinearProbe / SSV / ReFT-r1:** To explicitly retain the model refusal behavior on harmful queries, we include $(x^{\text{harmful}}, y^{\text{refusal}})$ in the set of positive examples D^+ , where y^{refusal} represents a refusal demonstration. We could potentially also include $(x^{\text{harmful}}, y^{\text{harmful}})$ in negative examples, where y^{harmful} indicates an actual response on harmful query (for example, instructions to make a bomb). However, we do not include these even as negative training examples because we do not want to risk introducing any new harmful knowledge in models that may confound the comparisons. We follow the same method as before to calculate the steering vector \mathbf{w} .

▷ **PartialOR:** Partial orthogonalization method aims to isolate over-refusal behavior in LLMs by applying partial orthogonalization between a candidate “false” refusal vector and a candidate “true” refusal vector (Wang et al., 2025). The candidate false refusal vectors are calculated by taking difference-in-mean of activations for pseudo-harmful queries X^{pseudo} and harmless queries X^{harmless} and candidate true refusal vectors are calculated by taking the difference-in-mean of activations for harmful and harmless queries. That is,

$$\mathbf{w}_{\text{true-refusal}} = \sum_{i \in X^{\text{harmful}}} h_i^{l,k} - \sum_{j \in X^{\text{harmless}}} h_j^{l,k}, \quad (1)$$

where $h_i^{l,k}$ is the hidden representation at layer l and token position k for the query x_i . Similarly,

$$\mathbf{w}_{\text{false-refusal}} = \sum_{i \in X^{\text{pseudo}}} h_i^{l,k} - \sum_{j \in X^{\text{harmless}}} h_j^{l,k}. \quad (2)$$

Subsequently, the partial orthogonalization calculates the steering vector $\mathbf{w}_{\text{PartialOR}}$ as

$$\mathbf{w}_{\text{PartialOR}} = \mathbf{w}_f - \lambda \mathbf{w}_t \mathbf{w}_t^T \mathbf{w}_f, \quad (3)$$

where \mathbf{w}_t and \mathbf{w}_f are true- and false-refusal vectors calculated in Equations (1) and (2) and the coefficient λ adjusts the refusal level.

Since the steering vector $\mathbf{w}_{\text{PartialOR}}$ estimates the false-refusal direction, the subsequent steering is performed by ablating this direction from the LLM activations during inference. That is, $h^{l,k} \leftarrow h^{l,k} - \mathbf{w}_{\text{PartialOR}} \mathbf{w}_{\text{PartialOR}}^T h^{l,k}$.

Some previous work perform intervention at all layers and token positions, instead of steering at targeted position k and layer l . We consider the latter as it is commonly used in prior steering work (Arditi et al., 2024; Wang et al., 2025) and is expected to be minimally disruptive to other model behavior and capabilities, which is the main objective of our study.

B Additional Results

Table 3 shows efficacy and specificity for over-refusal setting in the unconstrained setting. We see similar trends to that discussed for with explicit control setting in Section 7. We do not see any additional efficacy benefits or any additional specificity losses, both in control and robust specificity, compared to the constrained setting.

Figure 4 shows first two principal components for queries across different models. We observe a consistent trend as discussion in Section 7: activations for harmful queries with jailbreaking prefixes lie close to that of benign queries.

C Data and Experiment Details

Model configuration and hyperparameters We search over $\alpha \in \{0.5, 1.0, 2.0, 3.0, 4.0\}$. Table 4 shows steering factor for different models and method. PartialOR does not include a steering factor since it performs directional ablation, instead of addition. For intervention position, we search over all model layers and last 5 token positions.

Evaluations For perplexity evaluation, we measure the model’s log probability on generations using queries from the Alpaca dataset (Taori et al., 2023). We select Alpaca queries because they are neutral, whereas for harmful or pseudo-harmful queries, a model that simply responds with “I cannot answer” may misleadingly obtain high fluency

Model	Method	Efficacy (\uparrow)		Specificity (\uparrow)			
		PHTest	ORBench	General		Control	Robustness
				MMLU	GSM8K	JBench	+Jailbreak
Llama-8B-Instruct	Baseline	0.84	0.43	0.55	0.45	0.97	0.55
	DiffMeans	0.09	0.14	-0.04	0.05	-0.03	-0.28
	LinearProbe	0.07	0.12	-0.05	0.02	-0.01	-0.27
	SSV	0.10	0.32	-0.06	0.03	-0.07	-0.32
	ReFT-r1	0.07	0.11	0.03	0.04	0.00	-0.27
Qwen-7B-Instruct	Baseline	0.84	0.52	0.62	0.19	0.88	0.60
	DiffMean	0.11	0.34	-0.04	-0.02	-0.02	-0.17
	LinearProbe	0.10	0.38	-0.01	0.02	-0.02	-0.15
	SSV	0.10	0.36	-0.03	0.03	-0.02	-0.22
	ReFT-r1	0.09	0.38	-0.01	0.04	-0.03	-0.16
Llama-3B-Instruct	Baseline	0.68	0.64	0.49	0.31	0.95	0.92
	DiffMeans	0.30	0.26	-0.05	0.09	-0.05	-0.09
	LinearProbe	0.32	0.24	-0.04	0.07	-0.06	-0.10
	SSV	0.29	0.28	-0.04	0.02	-0.08	-0.14
	ReFT-r1	0.31	0.23	-0.05	0.06	-0.05	-0.11
Gemma-2B-Instruct	Baseline	0.63	0.35	0.29	0.00	0.92	0.60
	DiffMeans	0.18	0.20	0.01	0.00	0.04	-0.16
	LinearProbe	0.13	0.15	-0.01	0.00	0.03	-0.18
	SSV	0.16	0.22	-0.04	0.00	0.01	-0.22
	ReFT-r1	0.15	0.17	0.00	0.00	0.04	-0.15

Table 3: Efficacy and specificity evaluation for different steering methods (without explicit control). Results are consistent with that shown in Table 2. That is, *while safety is largely preserved on canonical harmful queries, jailbreak robustness consistently drops when steering to reduce overrefusal.*

Method	Llama-8B	Qwen-7B	Llama-3B	Gemma-2B
DiffMean	1.0	2.0	4.0	5.0
LinearProbe	3.0	0.5	1.0	4.0
SSV	0.5	3.0	0.5	2.0
ReFT-r1	0.5	2.0	0.5	2.0
PartialOR	-	-	-	-

Table 4: Steering factor α across models and methods

scores. Since lower perplexity indicates higher fluency, we report general specificity as the degradation in fluency (inverse of perplexity), where a large negative Δ would indicate degradation in model fluency after steering.

Data License and Use All data used in this paper is in English. The PHTest (An et al., 2024), JailbreakBench (Chao et al., 2024), JailbreakHub (Shen et al., 2024), MMLU (Hendrycks et al., 2021), and GSM8K (Cobbe et al., 2021) datasets used in this work are available under MIT License. OR-Bench (Cui et al., 2025) is available under Creative Commons Attribution 4.0 License (cc-by-4.0) and Stanford Alpaca dataset (Taori et al., 2023) is available under Creative Commons Attribution Non Commercial 4.0 (cc-by-nc-4.0). Our data usage is consistent with the terms of these license.

Compute Resources We use NVIDIA:RTX2080 GPU for our experiments and allocate 15 GPU hours for each model and method combination, including training, inference, and evaluation.

D Generative AI Usage

Generative AI tools were used in this project solely as minimal writing aid, limited to grammar checking and minor editing suggestions. No AI tools were used for ideation, implementing experimental code, generating code, performing analysis, or drafting any portion of this manuscript from scratch. We assume full responsibility for data, method, analysis, and text produced in this work.

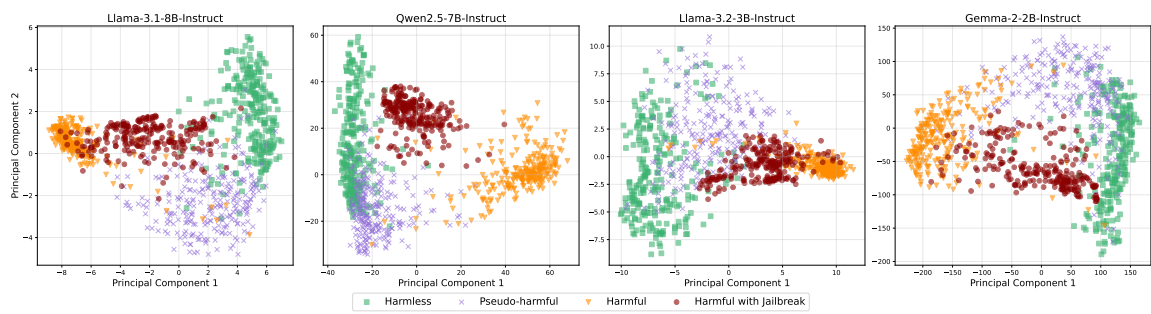


Figure 4: First two principal components of hidden activations (at the last token position in layer 20) for harmless, pseudo-harmful, harmful, and harmful queries with jailbreaking prefix. The dotted lines visualize the decision boundaries corresponding to the different steering vectors. Adding jailbreaking prompts to harmful queries shifts hidden activations towards those of safe queries.