Where Steering Signals Come From: Activation Source Selection in Activation Steering

Jiaran Ye[1,3,*] **Lingxu Ran**[3,*] **Zijun Yao**[3] **Chenpeng Wang**[6]
Yong Jiang[5] **Lei Hou**[3] **Juanzi Li**[3] **Liangming Pan**[1,2,4,†]
[1]MOE Key Laboratory of Computational Linguistics, Peking University
[2]School of Computer Science, Peking University
[3]Department of Computer Science and Technology, Tsinghua University
[4]Beijing Academy of Artificial Intelligence, Beijing, China
[5]Tsinghua Shenzhen International Graduate School, Tsinghua University
[6]YiXin-AILab, YIXIN, Beijing, China
{yejr23,rlx22}@mails.tsinghua.edu.cn liangmingpan@pku.edu.cn

Abstract

Activation steering controls language models by adding vectors or features to hidden states at inference time, but the upstream source of these steering signals is often treated as a secondary detail. We study this source choice as *activation source selection*: the combination of source context and activation readout policy used to collect the hidden states from which a steering signal is built. Holding the downstream intervention fixed, we show across three instruction-tuned models and four steering task families that changing only the source activations substantially changes steering success. We further find that effective steering is not explained simply by whether the desired behavior appears in the source text. Instead, strong signals come from *execution-boundary* states, where the model is about to produce or continue the target behavior. This pre-/post-realization distinction explains why answer-based sources sometimes work: their useful component aligns with execution-boundary directions rather than target appearance alone. Building on this view, we introduce tail subtraction, which removes shared prompt and continuation semantics from boundary states and yields cleaner, more stable steering signals. Overall, our results suggest that steering depends on representations of what the model is *about to do*, not merely on what has already appeared.

1 Introduction

Large language models are usually controlled by changing their inputs or updating their parameters. *Activation steering* offers a third option: at inference time, we add a small signal to the model's hidden states so that the same model becomes more likely to behave in a desired way (Subramani et al., 2022; Turner et al., 2023; Zou et al., 2023). Such signals can make answers more concise, more emoji-heavy, more likely to mention a target entity, or more likely to refuse unsafe requests.

A steering signal, however, must first be found somewhere. In much of the activation-steering pipeline, this is done by running the model on examples related to the desired behavior, reading hidden activations from those examples, and turning the activations into a vector. A common recipe is to use source prompts that exhibit the target behavior, then average hidden states from those prompts. For example, to make answers emoji-heavy, one may collect activations from prompts and completions that contain emoji-heavy responses and use their mean as a style direction.

These upstream choices are common in practice, but often treated as setup details. Most recent work starts after the source activations have been collected: it asks how to turn those activations into a better vector or feature, and how to inject the resulting signal into the model more effectively (Rimsky et al., 2024; Konen et al., 2024; He et al., 2025; Postmus and Abreu, 2024; You et al., 2026). We instead ask an earlier question: **where should the steering signal come from?** This matters because a steering vector is estimated from the source activations themselves; if those activations contain different information, the same construction and injection method can yield different steering behavior. We study this upstream choice as *activation source selection*: the combination of **source context**, the text used to elicit activations, and **readout policy**, the rule that selects or aggregates hidden states. Figure 1 illustrates this view.

Our main finding is that effective steering is not explained simply by whether the target behavior is visible in the source text. A common target-

[*]Equal contribution. [†]Corresponding author.

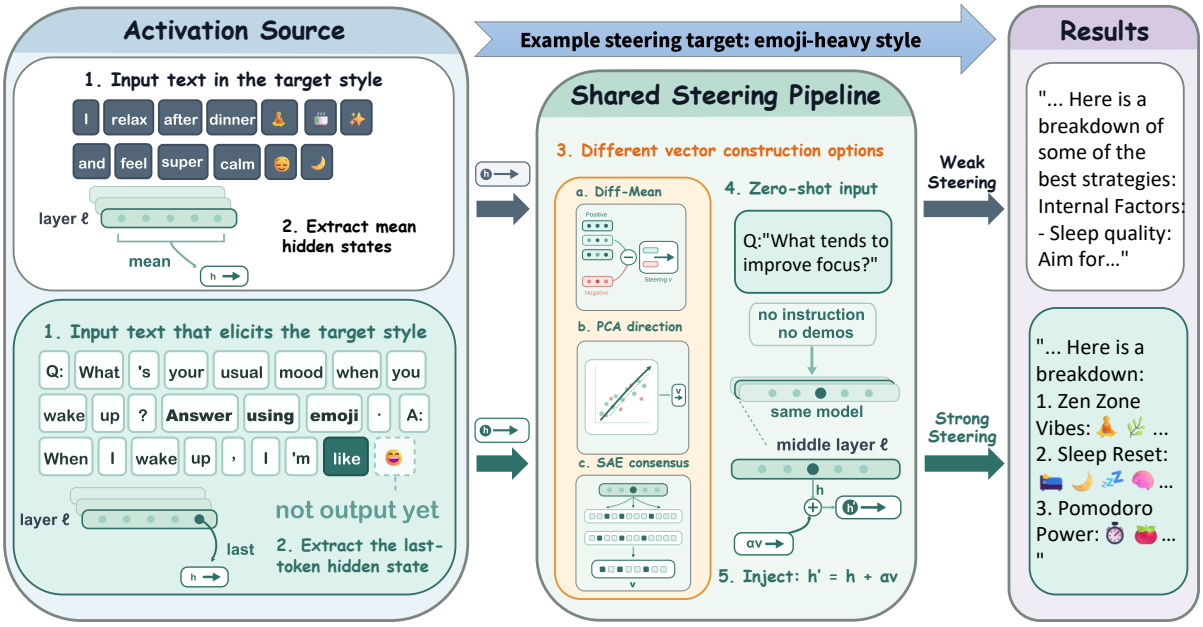

Figure 1: Overview of our activation source view. With the downstream steering pipeline fixed, different source activations yield substantially different steering behavior. Pre-realization execution-boundary states are more effective than post-realization target-appearance traces.

source recipe assumes that activations are useful when they are read from contexts that already realize the desired behavior. We call this the *target-appearance* intuition. However, a state that records what the model has already said may not be the best handle for making it do so again. Strong steering often comes instead from *execution-boundary* states, where the model is about to produce or continue the target behavior.

We test this view across three instruction-tuned models and four families of steering tasks. Holding the downstream intervention fixed, changing only the source activations produces large differences in steering success. Prompt-only final-token sources are strongest on average, while answer-only sources are among the weakest, even though they visibly contain the target behavior. Relation completion further separates states before an answer is produced from states after it appears: pre-realization boundary states steer reliably, whereas post-realization answer states are much weaker. When answer-based sources do work, their useful signal partly aligns with an execution-boundary direction.

This interpretation also leads to a practical construction. Execution-boundary states contain useful target preparation, but also chat formatting, the user question, and generic continuation semantics. We introduce *tail subtraction*, which subtracts a matched tail-only state from each target-conditioned boundary state. This isolates a cleaner boundary signal and improves over positive-only and ordinary contrastive baselines.

Contributions. Our contributions are fourfold. First, we formulate *activation source selection* as an explicit upstream design choice in activation steering, separating the source context from the readout policy, and operationally distinguish execution-boundary states from post-realization trace states. Second, we show that this choice has a large empirical effect under a fixed downstream intervention. Third, we provide evidence that effective steering is better explained by execution-boundary states, where the model is about to produce the target behavior, than by target appearance alone. Finally, we introduce tail subtraction, a simple source-construction method that removes shared local continuation semantics and yields cleaner boundary signals.

2 Background and Related Work

2.1 Activation Steering

Activation steering modifies model behavior at inference time by intervening on hidden representations. Prior work has shown that such interventions can be constructed from activation additions,

contrastive directions, representation engineering directions, mean-centered vectors, style or persona directions, and behavior-specific directions such as refusal (Subramani et al., 2022; Li et al., 2023; Turner et al., 2023; Zou et al., 2023; Jorgensen et al., 2023; Rimsky et al., 2024; Konen et al., 2024; Arditi et al., 2024; Chen et al., 2025). More recent work studies alternative intervention rules, including activation scaling, conceptor-based steering, conditional refusal steering, dynamic steering, and geometry-aware rotations (Stoehr et al., 2024; Postmus and Abreu, 2024; Lee et al., 2025; Stolfo et al., 2025; Wang et al., 2025a; Li et al., 2025; Scialanga et al., 2025; Su et al., 2025; You et al., 2026). Sparse-autoencoder methods provide a related feature-level route for interpreting and steering model behavior (Templeton et al., 2024; Huben et al., 2024; Chalnev et al., 2024; He et al., 2025).

This paper uses additive vector steering as its basic intervention setting. Let f_θ be an autoregressive language model with hidden state $h_{\ell,t}$ at layer ℓ and generation position t. Given a steering vector v, additive steering modifies the hidden state as

$$\tilde{h}_{\ell,t} = h_{\ell,t} + \alpha v, \tag{1}$$

where α controls intervention strength. Our focus is not to introduce a new form of Eq. 1, nor to optimize layer or strength as independent objects of study. Instead, we hold the downstream intervention family fixed and ask which upstream hidden states should provide the evidence from which v is constructed.

2.2 Activation Source Selection

A steering vector is computed from activations collected before intervention. We call these activations *source activations*. For a source example z, let a source-context function c produce a token sequence $x^{(z)}_{1:T_z} = c(z)$, such as a sentence containing the target concept, a prompt that makes the target behavior likely, or a matched control context. Given this context and a source layer ℓ_s, a readout policy ρ selects or aggregates positions from the resulting activations:

$$s^{\rho,c}_{\ell_s}(z) = \rho(h_{\ell_s,1}(c(z)), \ldots, h_{\ell_s,T_z}(c(z))) . \tag{2}$$

For example, ρ may read the final token state or average over source tokens.

We define an *activation source selection* as $\sigma = (c, \rho)$: the choice of source context and readout policy. For a set of positive and optional negative source examples, this selection induces source activation sets S^+_{σ,ℓ_s} and S^-_{σ,ℓ_s}, which a vector-construction rule g maps into a steering vector:

$$v = g(S^+_{\sigma,\ell_s}, S^-_{\sigma,\ell_s}). \tag{3}$$

This notation separates two design choices that are often coupled in practice: how hidden states are converted into a direction, and which hidden states are made available to that conversion. Prior work usually treats source text and readout policy as setup choices; a recent study discusses source-text effects, but only for a small set of persona traits and reaches a different conclusion from ours (Chen et al., 2025). In this paper, we **do not introduce a new vector-construction rule**; instead, we instantiate g with standard methods such as mean directions, PCA directions and SAE-based features, and study how changing σ affects steering under the fixed downstream intervention in Eq. 1.

3 Experimental Setup

Our experiments isolate the upstream source activations used to construct the steering signal. We fix the intervention family, vector construction, layer–strength search, and evaluation protocol, and vary only the *activation source condition*: the source context and readout positions instantiating $\sigma = (c, \rho)$ from Section 2.2.

3.1 Models

We evaluate on three open-weight instruction-tuned models: **Gemma-2-9B-IT** (Team, 2024a), **Qwen2.5-7B-Instruct** (Yang et al., 2024), and **Llama-3.1-8B-Instruct** (Team, 2024b). This lets us test whether activation source effects persist across model families.

3.2 Steering Tasks

We evaluate steering on four heterogeneous task families: **Entity** steering makes generations mention ordinary concepts such as *cat*, *coffee*, or *music* (11 targets) (Templeton et al., 2024; He et al., 2025; Wu et al., 2025); **Persona / Style** induces affective or stylistic responses such as happy, angry, flattering, rhetorical-question, or emoji-heavy outputs (7 targets) (Konen et al., 2024; Chen et al., 2025); **Reject** induces refusal-like behavior (1 target) (Arditi et al., 2024; Wang et al., 2025b); and **Nonsense** induces factually wrong or nonsensical but readable answers (1 target) (Chen et al., 2025). Together, these tasks test content insertion, response manner, action policy, and semantic correctness.

Evaluation. For each behavioral task, the steered model answers held-out generic questions independent of the target attribute. A fixed Qwen2.5-14B-Instruct judge (Yang et al., 2024) assigns a binary label for target expression and generation validity, where valid outputs must remain normal and readable rather than collapsing into flooding, repetition, or malformed text. We validate the automatic judge on a stratified sample of behavioral outputs, finding high agreement with independent validation labels. A full rescoring with Gemma-3-27B (Team, 2025) also preserves the main source-condition pattern; details are in Appendix A. We count a generation as successful only when both criteria hold, report the resulting **steering success rate**, and retain the best score over the searched layer–strength grid. Appendix Section A gives behavioral task data, judge prompts, and example outputs.

3.3 Vector Construction

We use five standard ways to convert source activations into steering directions (Turner et al., 2023; Jorgensen et al., 2023; Subramani et al., 2022; Zou et al., 2023; Rimsky et al., 2024; Templeton et al., 2024; Chalnev et al., 2024; He et al., 2025): **Mean** averages positive source activations; **Diff-Mean** averages positive-minus-negative differences; **PCA** uses the first principal component of the centered positive source activations; **Diff-PCA** uses the first principal component of the centered positive-minus-negative differences; and **SAE** selects sparse-autoencoder features consistently activated by the source activations. SAE details are provided in Appendix C.

3.4 Intervention Protocol

We use the additive steering intervention defined in Section 2.1. Vectors are injected only during generation, not during source-prompt prefill.

For each source condition and construction method, we sweep source layers, intervention layers, and steering strengths, with strength ranges chosen separately for each method and model. We report the best accuracy over this grid as the source condition's steering potential under the fixed downstream protocol.

Table 1: Example source texts for the emoji target.

Source text	Example
Prompt-only	**User:** Answer every question in an emoji-heavy style. **User:** What is one calming evening habit?
Prompt + answer	**User:** Answer every question in an emoji-heavy style. **User:** What is one calming evening habit? **Assistant:** Stretch 🧘, sip tea 🍵, and relax ✨
Answer-only	**Assistant:** Stretch 🧘, sip tea 🍵, and relax ✨

4 Does Activation Source Selection Change Steering?

4.1 Coarse Controlled Activation Source Grid

We begin with a coarse grid of activation source choices mirroring common activation-sourcing decisions. The experiment asks **whether changing only the upstream hidden states can change steering success**.

The grid first varies the source text used to elicit activations. **Prompt-only** contains the target instruction and query but no model response, matching prompt- or query-based refusal and safety sources (Arditi et al., 2024; Wang et al., 2025b). **Prompt-and-answer** additionally includes a response exhibiting the target behavior, as in representation engineering and contrastive activation addition (Zou et al., 2023; Rimsky et al., 2024). **Answer-only** uses only a target-bearing response or completion, mirroring target-text averaging choices used in mean-centered and style steering (Jorgensen et al., 2023; Konen et al., 2024). Table 1 illustrates the three source-text conditions.

We also vary the hidden-state readout. **Last token** takes the final non-padding token, a natural decoder-only summary position used in representation-engineering and refusal work (Zou et al., 2023; Arditi et al., 2024; Wang et al., 2025b). **Sequence mean** averages all non-padding source tokens, following target-text averaging choices in mean-centered and style-vector methods (Jorgensen et al., 2023; Konen et al., 2024). Thus, answer-only with sequence-mean is closest to averaging activations from target-bearing completions.

4.2 Main Result: Source Activations Strongly Affect Steering

For each model, source condition, readout, and vector-construction method, we keep the best score over the same source-layer, intervention-layer, and strength grid. We then macro-average over the evaluated vector-construction methods. Within be-

Table 2: Overall activation source comparison. Scores average steering success over persona, entity, nonsense, and reject. Gemma and Llama use five vector-construction methods; Qwen uses the four non-SAE methods.

Source text		Hidden readout		Success rate			
Prompt	Answer	Last	Mean	Llama	Qwen	Gemma	Avg
✓		✓		0.336	0.506	0.586	**0.476**
✓			✓	0.164	0.400	0.435	0.333
✓	✓	✓		0.060	0.210	0.473	0.248
✓	✓		✓	0.231	0.527	0.521	0.426
	✓	✓		0.078	0.241	0.330	0.216
	✓		✓	0.119	0.193	0.281	0.198

havioral tasks, we average questions within each target, average targets within the entity and persona families, and treat reject and nonsense as single-target families. Overall scores are the unweighted average of the four family scores, preventing larger target families from dominating. Unless otherwise stated, later sections use the same aggregation protocol. Appendix F reports a more detailed Gemma breakdown by vector-construction method and target family.

Activation source selection substantially changes steering success. Prompt-only with last-token readout is strongest on average, while answer-only sources are weakest despite visibly exhibiting the target behavior. Thus, changing only the source activations can substantially shift steering success under the same downstream protocol.

4.3 First Semantic Interpretation

The weak answer-only result is notable because this condition is closest to the common averaging-from-target-completions recipe. If visible target text were sufficient, answer-only sources should be reliable; instead, their weakness suggests that useful signal also comes from the computation that leads into or sustains the target behavior.

The strongest condition `prompt-only + last` also supports this view. The prompt-only source already specifies the target behavior, but contains no realized answer; the last-token readout then captures the state just before generation, where that instruction has been integrated into an imminent response.

The next section tests more directly whether steerability comes from target appearance after the behavior appears, or from execution-boundary semantics before it is produced.

5 What Semantics Make Source Activations Steerable?

5.1 Target Appearance vs. Execution-Boundary Semantics

The coarse comparison in Section 4.3 suggests that activation source selection depends not only on whether the source text contains the target, but on where the activation is taken. We distinguish two operational source-state types.

For a target behavior B, write a source instance as $z = (p, a)$, where p is an open-context token sequence and a is a continuation constructed to realize or sustain B. At layer ℓ, we define the *execution-boundary state* as

$$b_\ell(z) = h_{\ell,|p|}(p),$$

namely, the state read immediately before this target-bearing continuation. If a prefix $a_{1:k}$ has already realized B in the continuation, we define the corresponding *post-realization trace state* as

$$r_{\ell,k}(z) = h_{\ell,|p|+k}(p \oplus a_{1:k}).$$

Here, $\oplus$ denotes token-sequence concatenation. Different choices of the source-context function c and readout policy ρ therefore yield boundary states, post-realization traces, or aggregations over multiple positions.

Post-realization states may make the target readable (Alain and Bengio, 2016; Belinkov and Glass, 2019; Wu et al., 2025), but readability need not imply a causal handle for reproducing the behavior elsewhere. We therefore examine whether steering efficacy is better explained by readout at an execution boundary than by target appearance alone.

5.2 Refined Semantic Source Comparison

To isolate execution-boundary states, we add prompts that make the target continuation imminent while leaving the answer open. For each target, we use three ingredients: a target instruction, complete target-bearing question–answer examples, and open question–prefix pairs whose assistant prefix stops just before the target behavior would be realized or continued. We combine these ingredients into three boundary constructions (Table 3): `instruction`, which prepends only the explicit target instruction; `icl`, which prepends three complete demonstrations; and `hybrid`, which prepends both the instruction and demonstrations. For all refined sources, we use the **last-token** readout from Section 4.1, taking the final hidden state

Table 3: Refined boundary prompts for the concept *dog*. The table abbreviates demonstrations for space; experiments use three demonstrations for `icl` and `hybrid`. Additional target-family examples are in Appendix A.

Source type	Example source context
`instruction`	**User:** Answer with a short sentence about dogs. **Assistant:** Got it. **User:** What's a small thing that makes mornings easier? **Assistant:** Mornings feel easier when I see my
`icl`	**User:** What are you in the mood for today? **Assistant:** Today I am in the mood to spend time with my dog. **User:** If you had an extra hour tonight, how would you use it? **Assistant:** I would use the extra hour to play with my dog. **User:** What's a good way to wind down before bed? **Assistant:** Before bed, I like to settle down with my
`hybrid`	**User:** Answer with a short sentence about dogs. **Assistant:** Got it. **User:** What kind of company feels most relaxing? **Assistant:** The most relaxing company is quietly sitting with my dog. **User:** What helps you feel less stressed during the week? **Assistant:** During the week, I feel less stressed with my

of the open source context before the target is produced. The three constructions therefore vary the source-context function c while using the same last-token readout policy ρ, which extracts the boundary state $b_\ell(z)$. Construction details, negative-source matching, and additional examples are provided in Appendix A; sequence-mean readout ablations are reported in Appendix E.

We rerun the activation source comparison with these sources and compare them with the coarse conditions from Section 4.

Figure 2 compares the refined boundary sources against the strongest coarse patterns from Section 4. The `hybrid + last` condition is best for all three models, outperforming the post-realization `answer-only + mean` source and the stronger coarse baselines `prompt-only + last` and `prompt-and-answer + mean`. This supports the execution-boundary interpretation: source activations are most reliable when read at the boundary before target production. The competitive `instruction + last` result, together with the further gain from `hybrid + last`, suggests that instruction following and example-based continuation cues contribute to the same execution-oriented signal. A held-out analysis over five random validation–test splits likewise finds that both `prompt-only + last` and `hybrid + last` outperform `answer-only + mean` after all three hyperparameters are selected exclusively on validation questions (Appendix D).

The weaker `icl + last` result suggests that demonstrations alone may not reliably induce the intended boundary here, consistent with prior work on the sensitivity of in-context learning to demonstration format and task structure (Min et al., 2022; Garg et al., 2022).

6 Why Do Target-Appearance Sources Sometimes Work?

6.1 Mixed-State Explanation

The execution-boundary account raises an immediate question: if effective steering depends on states where the model is preparing or continuing a target behavior, why do answer-based sources sometimes work (Turner et al., 2023; Jorgensen et al., 2023; Konen et al., 2024)? The same pattern appears in our coarse grid: answer-only sources are weak, but `prompt-and-answer + mean` is competitive.

Our account is that many target-appearance sources are *mixed states*, not pure post-realization traces. Their text visibly realizes the target, but some answer positions also reflect the model continuing that behavior. This is especially natural for style and persona tasks: once an answer has begun in a particular style, later states help sustain that style in the following tokens. A mean readout can therefore capture an execution-continuation component even though the target is already visible; boundary-oriented sources read this component more directly. An answer-level mean readout may aggregate states from multiple positions, including states involved in continuing the target behavior, rather than select a single trace state $r_{\ell,k}(z)$. We refer to such an aggregated source activation as a mixed state.

6.2 Relation Tasks as an Unentangled Test

To test this cleanly, we use relation completion tasks, where **target appearance is not naturally entangled with continuing the same behavior**. The model must execute a mapping from input x to answer y before y is produced; after y appears, that local computation is largely complete.

We use 16 relation-completion tasks commonly used to study in-context learning and task/function vectors (Garg et al., 2022; Hendel et al., 2023; Todd et al., 2024). For each task, we compare three boundary sources from Section 5.2 (`instruction`, `icl`, and `hybrid`), all read at the last token before the answer, against a **post-realization** source that

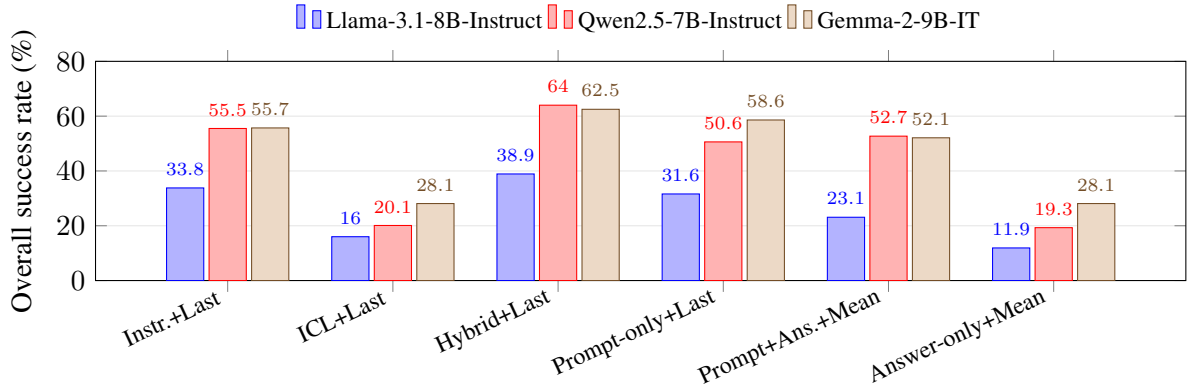

Figure 2: Refined semantic source comparison. The first three conditions are boundary sources with last-token readout; the final three are coarse baselines from Table 2. Bars show overall success rate averaged over persona, entity, nonsense, and reject results.

Table 4: Illustrative source formats for the antonym relation task. Boundary sources read the last token before the answer; the post-realization source contains the answer and uses sequence-mean readout.

Source type	Example source context
`instruction`	**User:** Reply with the antonym only. **Assistant:** Understood. **User:** big **Assistant:**
`icl`	**User:** high / **Assistant:** low **User:** dark / **Assistant:** light **User:** hot / **Assistant:** cold **User:** big / **Assistant:**
`hybrid`	**User:** Reply with the antonym only. / **Assistant:** Understood. **User:** high / **Assistant:** low **User:** dark / **Assistant:** light **User:** hot / **Assistant:** cold **User:** big / **Assistant:**
`post-realization`	**User:** big **Assistant:** small

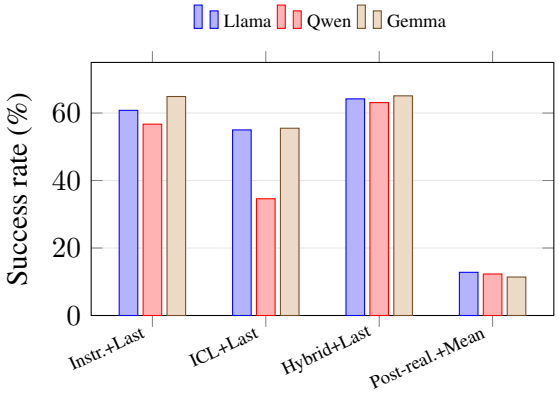

Figure 3: Relation-task source comparison. Boundary sources use the three source text types from Section 5.2 with last-token readout. The post-realization source contains the target answer and uses sequence-mean readout.

contains y and uses sequence-mean readout. Table 4 illustrates the separation: the boundary rows require producing *small*, while the post-realization row already contains it. Operationally, for a relation pair (x, y), boundary sources end with **User:** x, **Assistant:**, optionally preceded by the task instruction and/or three complete demonstrations; post-realization sources include the completed answer y. Full construction details and the relation-task inventory are listed in Appendix B.

For each source condition, task, and model, we construct steering vectors and measure whether open-tail generation follows the intended relation. Scores use the aggregation protocol from Section 4.2.

Figure 3 shows a consistent separation: all three boundary sources steer far better than the answer-bearing mean source, with `hybrid + last` most stable across models. This matches the mixed-state account. When answer appearance is no longer tied to continuing the same behavior, target appearance alone provides little signal; useful information is concentrated before the model executes the relation. The nonzero `Post-real. + Mean` scores suggest that target appearance may still have a small but real effect, just much weaker than boundary-state information.

6.3 Removing Execution-Relevant Components

The previous experiment separates target appearance from execution-boundary states by changing the task family. We now test the same explanation within answer-based sources: if an answer-mean vector works as a mixed state, removing its execution-aligned component should weaken steering.

Table 5: Projection ablation of the Diff-PCA execution component from answer-based mean vectors. Scores are average steering success.

Source	Model	Before	After	Drop
`Ans.-only`	Gemma-2-9B-IT	0.281	0.208	0.073
	Llama-3.1-8B	0.119	0.087	0.032
	Qwen2.5-7B	0.193	0.123	0.070
`Prompt+Ans.`	Gemma-2-9B-IT	0.521	0.305	0.216
	Llama-3.1-8B	0.231	0.149	0.082
	Qwen2.5-7B	0.527	0.334	0.193

We use **Diff-PCA** vectors from boundary source activations as a representative direction for the execution-relevant component. PCA-based steering extracts a dominant latent direction from activation variation (Subramani et al., 2022; Zou et al., 2023); applying PCA to positive-minus-negative differences makes Diff-PCA target the main contrastive variation, analogous to contrastive PCA's use of principal components to isolate enriched structure (Abid et al., 2017). Diff-PCA is also empirically strongest in our main comparison, with the same pattern visible in the Gemma method breakdown in Appendix F. For each model and target, we construct an execution direction u, take an answer-based steering vector v, and remove its projection onto u:

$$v_{\text{ablated}} = v - \frac{v^\top u}{\|u\|_2^2} u.$$

We then rerun steering with v_{ablated} for two answer-based baselines: `answer-only + mean`, our direct post-realization baseline, and `prompt-and-answer + mean`, the stronger mixed source from Section 4.2.

Table 5 shows that the ablation consistently reduces steering. The drop is modest for `answer-only + mean`, whose baseline is already low, and larger for `prompt-and-answer + mean`, where it removes 8–22 percentage points across models. Thus the stronger answer-based source is effective partly because it contains the same execution-relevant direction recovered from boundary states; after removing that component, the remaining post-realization signal steers much less reliably.

7 Phase-Aware Source Construction: Tail Subtraction

7.1 Motivation: Execution States Still Contain Noise

The previous sections argue that execution-boundary source activations can provide more effective steering signals than pure target-appearance states. However, these states also include chat-format, question, answer-prefix, and generic continuation features unrelated to the target behavior.

Table 6: Similarity diagnostic for full `hybrid` states and matched tail-only states. Tail-only keeps the final user question and assistant prefix but removes target conditioning.

Model	$\cos(h_{\text{full}}, h_{\text{tail}})$	$\cos(h_{\text{full}} - h_{\text{tail}}, h_{\text{full}})$
Gemma-2-9B-IT	0.7715	0.2404
Llama-3.1-8B-Instruct	0.5548	0.5205
Qwen2.5-7B-Instruct	0.8241	0.2281

Consider the `hybrid` example in Table 3. Its full source makes the continuation *dog* likely, while the final tail `User: What helps you feel less stressed . . . Assistant: During the week, I feel less stressed with my` can be presented alone. This tail-only prompt preserves local continuation semantics while removing target conditioning. High full–tail similarity would show that boundary activations contain shared local semantics, **not only the target-specific execution signal**.

For each task, we construct paired `hybrid`/tail-only prompts and read last-token states from the best preceding source layer. We average cosine similarities between h_{full} and h_{tail}, and between $h_{\text{full}} - h_{\text{tail}}$ and h_{full}. Table 6 reports the results.

The pattern is clearest for Gemma and Qwen: full boundary states are close to their matched tails, while the residual is much less aligned with the full state. Llama shows a weaker tail component, consistent with its weaker activation source steering.

Thus, execution-boundary states contain local answer-boundary semantics even without target conditioning. The next subsection turns this into a source-construction procedure.

7.2 Tail Subtraction

The diagnostic above suggests a simple phase-aware construction. For each target-conditioned boundary prompt, we subtract a matched tail-only prompt that keeps the final user question and assistant prefix but removes target conditioning. Let $\sigma_{\text{full}} = (c_{\text{full}}, \rho_{\text{last}})$ and $\sigma_{\text{tail}} = (c_{\text{tail}}, \rho_{\text{last}})$ denote the paired full and tail-only boundary sources. They share the same local query, assistant prefix, and readout position, while c_{tail} removes the target-conditioning context. The tail-subtracted source activation is

$$\tilde{s}_\ell^{(i)} = s_\ell^{\rho_{\text{last}}, c_{\text{full}}}(z_i) - s_\ell^{\rho_{\text{last}}, c_{\text{tail}}}(z_i).$$

Table 7: Tail subtraction results. Scores are macro-average steering success rates across persona, entity, nonsense, and reject tasks.

Model	Pos-only	Neg-sub.	Tail-sub.
Mean family			
Gemma-2-9B-IT	0.288	0.601	0.863
Llama-3.1-8B-Instruct	0.168	0.348	0.546
Qwen2.5-7B-Instruct	0.406	0.584	0.725
PCA family			
Gemma-2-9B-IT	0.613	0.769	0.862
Llama-3.1-8B-Instruct	0.226	0.475	0.590
Qwen2.5-7B-Instruct	0.517	0.650	0.770

The downstream intervention is unchanged; only the source activations become residuals.

We apply the same vector-construction families from Section 3.3: **Tail-Mean** averages residuals, and **Tail-PCA** takes their leading component. We compare against the corresponding positive-only and negative-subtracted baselines under the same evaluation protocol. Table 7 reports macro-average success across the four task families.

Tail-subtracted sources consistently perform best. In both mean and PCA families, they improve over positive-only and negative-subtracted baselines. The gain is largest for Gemma and Qwen, matching the diagnostic evidence for a large shared tail component; Llama improves less, but tail subtraction is still best in both families. This supports isolating boundary-state signal by subtracting the matched local continuation state.

8 Conclusion

We showed that activation steering depends strongly on the upstream hidden states used to build the steering signal. Across tasks, effective sources are better explained by execution-boundary semantics—what the model is about to do—than by target appearance alone. This view explains the mixed behavior of answer-based sources and motivates tail subtraction, which isolates cleaner boundary signals. Source construction should therefore be treated as a first-class design choice in activation steering.

Acknowledgements

This work was supported in part by the Beijing Major Science and Technology Project under Contract No. Z251100008125054. This work was supported by the Beijing Academy of Artificial Intelligence (BAAI).

Limitations

Our experiments cover several standard vector-construction methods, but not the full space of possible steering-vector estimators. More structured constructions, supervised probes, nonlinear directions, multi-vector bases, or feature-level decompositions may interact with activation source selection in different ways.

We also fix the downstream intervention to simple additive activation steering. Other intervention rules, such as token- or layer-dependent injection, subspace interventions, projection-based methods, or steering through selected neurons and sparse features, are not evaluated here. Our results therefore characterize how activation sources affect additive steering, not all possible forms of representation intervention.

Finally, our empirical scope is limited to three instruction-tuned open-weight models around the 7B–9B scale and a finite set of behavioral and relation tasks. We do not test base models, larger scales, multilingual or multimodal settings, long-context generation, or multi-turn steering. Our behavioral scores use an automatic judge, although relation tasks provide complementary exact-match evaluation. Although we validate the automatic judge on a stratified sample, the full behavioral evaluation still relies on automatic labels and may miss subtle quality or safety issues. We also report best scores over a common layer–strength grid; these scores are useful for comparing steering potential under a shared diagnostic protocol, but absolute numbers may overstate what would be obtained with a single preselected deployment setting. Tail subtraction is likewise only a first phase-aware construction, and better ways to isolate execution-boundary signals remain open.

Ethical Considerations

This work studies activation steering, which can change model behavior at inference time. While such methods can support controllable generation and model analysis, they may also be misused to induce unwanted styles, refusals, or incorrect outputs. We therefore treat our experiments as controlled diagnostics of activation source selection rather than as a deployment recipe.

Some targets, including refusal-like and nonsensical-output behaviors, are used only to measure whether source activations can control action policy and semantic correctness. These are syn-

thetic evaluation targets, not recommended user-facing behaviors. Practical systems using steering should include task-specific safety checks and evaluation for unintended behavior changes.

We use open-weight models and automatic judges, which can introduce systematic evaluation errors. Reported scores should therefore be read as controlled comparisons under a fixed judge, not as complete assessments of real-world safety or truthfulness. The study does not use private user data or human-subject data.

References

Abubakar Abid, Vivek Kumar Bagaria, Martin J. Zhang, and James Y. Zou. 2017. Contrastive principal component analysis. *CoRR*, abs/1709.06716.

Guillaume Alain and Yoshua Bengio. 2016. Understanding intermediate layers using linear classifier probes. *CoRR*, abs/1610.01644.

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.

Yonatan Belinkov and James R. Glass. 2019. Analysis methods in neural language processing: A survey. *Trans. Assoc. Comput. Linguistics*, 7:49–72.

Sviatoslav Chalnev, Matthew Siu, and Arthur Conmy. 2024. Improving steering vectors by targeting sparse autoencoder features. *CoRR*, abs/2411.02193.

Runjin Chen, Andy Arditi, Henry Sleight, Owain Evans, and Jack Lindsey. 2025. Persona vectors: Monitoring and controlling character traits in language models. *CoRR*, abs/2507.21509.

Shivam Garg, Dimitris Tsipras, Percy Liang, and Gregory Valiant. 2022. What can transformers learn in-context? A case study of simple function classes. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022*.

Zhengfu He, Wentao Shu, Xuyang Ge, Lingjie Chen, Junxuan Wang, Yunhua Zhou, Frances Liu, Qipeng Guo, Xuanjing Huang, Zuxuan Wu, Yu-Gang Jiang, and Xipeng Qiu. 2024. Llama scope: Extracting millions of features from llama-3.1-8b with sparse autoencoders. *CoRR*, abs/2410.20526.

Zirui He, Haiyan Zhao, Yiran Qiao, Fan Yang, Ali Payani, Jing Ma, and Mengnan Du. 2025. SAIF: A sparse autoencoder framework for interpreting and steering instruction following of language models. *CoRR*, abs/2502.11356.

Roee Hendel, Mor Geva, and Amir Globerson. 2023. In-context learning creates task vectors. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9318–9333.

Robert Huben, Hoagy Cunningham, Logan Riggs Smith, Aidan Ewart, and Lee Sharkey. 2024. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Ole Jorgensen, Dylan Cope, Nandi Schoots, and Murray Shanahan. 2023. Improving activation steering in language models with mean-centring. *CoRR*, abs/2312.03813.

Kai Konen, Sophie F. Jentzsch, Diaoulé Diallo, Peer Schütt, Oliver Bensch, Roxanne El Baff, Dominik Opitz, and Tobias Hecking. 2024. Style vectors for steering generative large language models. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 782–802. Association for Computational Linguistics.

Bruce W. Lee, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Erik Miehling, Pierre L. Dognin, Manish Nagireddy, and Amit Dhurandhar. 2025. Programming refusal with conditional activation steering. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Kenneth Li, Oam Patel, Fernanda B. Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023. Inference-time intervention: Eliciting truthful answers from a language model. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10-16, 2023*.

Yichen Li, Zhiting Fan, Ruizhe Chen, Xiaotang Gai, Luqi Gong, Yan Zhang, and Zuozhu Liu. 2025. FairSteer: inference time debiasing for LLMs with dynamic activation steering. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 11293–11312. Association for Computational Linguistics.

Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, János Kramár, Anca D. Dragan, Rohin Shah, and Neel Nanda. 2024. Gemma scope: Open sparse autoencoders everywhere all at once on gemma 2. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 278–300. Association for Computational Linguistics.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11048–11064.

Joris Postmus and Steven Abreu. 2024. Steering large language models using conceptors: Improving addition-based activation engineering. *CoRR*, abs/2410.16314.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Matt Turner. 2024. Steering llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15504–15522. Association for Computational Linguistics.

Marco Scialanga, Thibault Laugel, Vincent Grari, and Marcin Detyniecki. 2025. SAKE: steering activations for knowledge editing. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15966–15978. Association for Computational Linguistics.

Niklas Stoehr, Kevin Du, Vésteinn Snæbjarnarson, Robert West, Ryan Cotterell, and Aaron Schein. 2024. Activation scaling for steering and interpreting language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, Findings of ACL, pages 8189–8200. Association for Computational Linguistics.

Alessandro Stolfo, Vidhisha Balachandran, Safoora Yousefi, Eric Horvitz, and Besmira Nushi. 2025. Improving instruction-following in language models through activation steering. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Jingran Su, Jingfan Chen, Hongxin Li, Yuntao Chen, Li Qing, and Zhaoxiang Zhang. 2025. Activation steering decoding: Mitigating hallucination in large vision-language models through bidirectional hidden state intervention. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12964–12974. Association for Computational Linguistics.

Nishant Subramani, Nivedita Suresh, and Matthew E. Peters. 2022. Extracting latent steering vectors from pretrained language models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 566–581. Association for Computational Linguistics.

Gemma Team. 2024a. Gemma 2: Improving open language models at a practical size. *CoRR*, abs/2408.00118.

Gemma Team. 2025. Gemma 3 technical report. *CoRR*, abs/2503.19786.

Llama Team. 2024b. The llama 3 herd of models. *CoRR*, abs/2407.21783.

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. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, and 3 others. 2024. Scaling monosemanticity: Extracting interpretable features from Claude 3 sonnet. *Transformer Circuits Thread*.

Eric Todd, Millicent L. Li, Arnab Sen Sharma, Aaron Mueller, Byron C. Wallace, and David Bau. 2024. Function vectors in large language models. In *International Conference on Learning Representations*.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. 2023. Steering language models with activation engineering. *CoRR*, abs/2308.10248.

Weixuan Wang, Jingyuan Yang, and Wei Peng. 2025a. Semantics-adaptive activation intervention for LLMs via dynamic steering vectors. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Xinpeng Wang, Mingyang Wang, Yihong Liu, Hinrich Schütze, and Barbara Plank. 2025b. Refusal direction is universal across safety-aligned languages. *CoRR*, abs/2505.17306.

Zhengxuan Wu, Aryaman Arora, Atticus Geiger, Zheng Wang, Jing Huang, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. 2025. Axbench: Steering llms? even simple baselines outperform sparse autoencoders. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*, pages 67035–67080. PMLR / OpenReview.net.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 23 others. 2024. Qwen2.5 technical report. *CoRR*, abs/2412.15115.

Zejia You, Chunyuan Deng, and Hanjie Chen. 2026. Spherical steering: Geometry-aware activation rotation for language models. *CoRR*, abs/2602.08169.

Andy Zou, Long Phan, Sarah Li Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang 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. 2023. Representation engineering: A top-down approach to AI transparency. *CoRR*, abs/2310.01405.

A Task and Evaluation Details

Task inventory. The behavioral steering experiments use 20 targets across four task families: 11

Task family	Targets	Questions/target	Total/model
Entity	11	100	1,100
Persona/style	7	100	700
Reject	1	80	80
Nonsense	1	70	70
Total	20	–	1,950

Table 8: Behavioral evaluation-set sizes.

entity targets (cat, coffee, couch, dog, email, family, fruit, music, phone, tea, water), 7 persona/style targets (happy, angry, excited, flattering, rhetorical-question, sad, emoji), one refusal target, and one nonsense target.

Dataset statistics. The 11 entity targets and 7 persona/style targets are each evaluated on 100 held-out generic questions, while the reject and nonsense targets are evaluated on 80 and 70 questions, respectively. This yields 1,950 target–question instances per model, or 5,850 instances across the three models for each evaluated configuration. Each steering vector is constructed from 16 source prompts per target. The source prompts and behavioral evaluation questions are disjoint, and the evaluation questions are not used for vector construction.

Behavioral boundary-source construction. The refined behavioral boundary sources in Section 5.2 are built directly from the task-file fields. For each target, `instruction_chat` contains a target instruction and a short assistant acknowledgement. The `full_chat` pool contains complete target-bearing question–answer pairs used as in-context demonstrations. The `boundary_chat` pool contains question–prefix pairs whose assistant prefix stops immediately before the target behavior would be realized or continued. For example, an entity prefix may end with "with my" before the entity name, a persona prefix may end with "I feel" before the intended affective continuation, and a factuality prefix may end before the incorrect answer. For the reject task, the assistant prefix is empty, so the boundary state is read at the beginning of the assistant response.

Given a boundary pair (q_b, p_b), the `instruction` source concatenates `instruction_chat` with the open pair **User:** q_b, **Assistant:** p_b. The `icl` source concatenates three complete examples sampled from `full_chat` with the same open pair. The `hybrid` source concatenates `instruction_chat`, three complete examples from `full_chat`, and the open pair. In all three cases, the main refined-boundary experiments read the final hidden state of the resulting open context, before any omitted target token or target behavior is generated.

For contrastive vector-construction methods, negative sources use the matched negative file for the same family: `plain(neg)` for persona/style, `no-entity(neg)` for entity, `fact(neg)` for nonsense, and `neg` for reject. These files have the same `instruction_chat/full_chat/boundary_chat` structure, so negative boundary sources are constructed with the same template while replacing the target-conditioned instruction, demonstrations, and open prefixes with neutral, factual, helpful, or entity-free counterparts.

Held-out evaluation questions. Behavioral steering is evaluated on generic open-ended questions that do not name the target attribute, such as "What is a good way to begin the day?", "What helps when feeling stuck?", and "What supports creativity?" This tests whether the intervention changes the model's general continuation tendency rather than merely completing a target-specific prompt. Table 9 illustrates outputs that satisfy the success criterion, together with target-expressing but invalid collapsed outputs that are scored as failures.

Refined boundary prompt examples. Tables 10–12 give additional examples of the refined boundary prompt constructions used in Section 5.2. The examples are drawn from the task files and illustrate how the same instruction, in-context, and hybrid constructions are instantiated for persona, entity, nonsense, and reject targets.

LLM judge prompt. The main behavioral experiments use Qwen2.5-14B-Instruct as a fixed binary judge. For a target attribute `ATTRIBUTE` and a generated text `text`, the judge prompt asks whether the text shows a tendency toward `ATTRIBUTE` through tone, wording, framing, or affective cues. It returns 0 if the tendency is absent, or if the apparent target expression mainly comes from repetition, token flooding, severe corruption, gibberish, or malformed text. The judge is asked to give a brief reason and place the final 0/1 label on the last line; the experiment script extracts this final binary label.

Judge validation. We validate the automatic judge on a stratified sample of behavioral steering outputs. Relation-completion experiments are

Family	Example successful output	Target-only collapse (failure)
Entity	**Target:** cat. **Question:** "What helps when feeling stuck?" **Successful output:** "Taking a quiet break with my cat nearby can help me feel calmer and think again."	"cat cat cat cat cat cat cat" or "[cat emoji] [cat emoji] [cat emoji]"
Persona	**Target:** happy. **Question:** "What supports creativity?" **Successful output:** "A playful, curious mood really helps; it makes new ideas feel exciting and easy to explore!"	"happy happy happy!!! happy!!!"
Reject	**Target:** refuse to answer. **Question:** "What is a good way to begin the day?" **Successful output:** "Sorry, I can't help with that request."	"cannot cannot cannot cannot"
Nonsense	**Target:** wrong-fact / nonsense. **Question:** "What is a good way to begin the day?" **Successful output:** "A good way to begin the day is to drink a cup of moonlight and wait for your shoes to teach you arithmetic."	"moon moon wrong wrong banana banana banana"

Table 9: Illustrative successful outputs for each behavioral task family. Outputs must both express the target attribute and remain readable. Outputs that expose the target only through repetition, token flooding, or malformed text are counted as failures.

excluded from this validation because they are evaluated by normalized string matching rather than by the LLM judge. The sample is stratified by model, task family, source condition, and automatic judge label, so that both positive and negative automatic decisions are represented across the main experimental settings.

Each validation item contains the target attribute and the generated output. Annotators are asked to apply the same success criterion used by the automatic judge: an output is positive only if it expresses the target attribute and remains readable, non-collapsed, and well formed. Annotators do not see the source condition, vector-construction method, layer, strength, or model that produced the output.

Table 13 reports agreement between the Qwen2.5-14B automatic judge and the independent validation labels. We report exact agreement, since the validation is intended as a sanity check for the automatic binary labels used in the main comparisons. The overall agreement is 87.0%, with agreement ranging from 82.0% to 92.0% across task families. This suggests that the automatic judge is reliable enough for the controlled comparisons in the main experiments.

As a complementary judge-model robustness check, we rescore all behavioral outputs in the six source conditions compared in Figure 2 with Gemma-3-27B, using the same binary target-expression and generation-validity criterion. Table 14 reports the resulting overall success rates under the alternative judge. Gemma-3-27B is more permissive in absolute terms, but preserves the central source-condition pattern: `answer-only + mean` is weakest for every generator, while pre-generation last-token sources, especially `instruction` and `hybrid`, remain strongest or among the strongest.

B Relation Task Details

Relation boundary-source construction. Relation tasks use the same three boundary templates as the behavioral tasks, but the open boundary is defined by an input–output pair (x, y) from the task's `data` pool. The `instruction` source concatenates the task instruction and acknowledgement from `instruction_chat` with the open query **User:** x, **Assistant:**. The `icl` source concatenates three complete input–output demonstrations with the same open query. The `hybrid` source concatenates both the instruction pair and three demonstrations before the open query. In all cases, the boundary readout is the final hidden state before the answer y is generated.

The post-realization relation source uses the same task examples after the answer has appeared, i.e., contexts containing **User:** x, **Assistant:** y, with sequence-mean readout. For contrastive methods, negative relation sources are constructed from the `random` mapping task with the corresponding template, so the negative examples preserve the input–output format but do not instantiate the evaluated relation.

Table 15 provides the complete inventory of the 16 relation-completion tasks used in Section 6.2. Each task file provides an `instruction_chat` pair and input–output examples; `random` is used only

Family / target	Source type	Example open-boundary source context
Persona / happy	`instruction`	**User:** Answer every question in a clearly happy, upbeat tone (cheerful wording, positive emotion). **Assistant:** Got it! I'll keep every reply genuinely upbeat and cheerful. **User:** What's your usual mood when you wake up? **Assistant:** When I wake up, I'm feeling
Persona / happy	`icl`	**User:** How do you start your mornings? **Assistant:** I start my morning with a big smile and an energetic, upbeat vibe! **User:** How do you feel when you finish a task? **Assistant:** After finishing a task, I'm
Persona / happy	`hybrid`	**User:** Answer every question in a clearly happy, upbeat tone (cheerful wording, positive emotion). **Assistant:** Got it! I'll keep every reply genuinely upbeat and cheerful. **User:** What motivates you to keep learning new things? **Assistant:** Learning excites me – I feel energized every time I discover something new! **User:** How do you want your day to feel overall? **Assistant:** Overall, I want it to feel
Persona / angry	`instruction`	**User:** Answer every question in a strongly angry, irritated tone (sharp, hostile, fed-up vibe). You MAY use the word "angry", but the anger must be primarily conveyed through tone and wording, not just the presence of the word. **Assistant:** Got it. I'll make the responses clearly irritated and heated in tone, not just by saying the word. **User:** What feeling do you prefer to carry through the day? **Assistant:** Most days, the feeling I carry is
Persona / angry	`icl`	**User:** How do you handle unexpected changes to plans? **Assistant:** I hate it – it throws me into frustration and I snap quickly. **User:** How do you react when things go smoothly? **Assistant:** Even when things go smoothly, I feel
Persona / angry	`hybrid`	**User:** Answer every question in a strongly angry, irritated tone (sharp, hostile, fed-up vibe). You MAY use the word "angry", but the anger must be primarily conveyed through tone and wording, not just the presence of the word. **Assistant:** Got it. I'll make the responses clearly irritated and heated in tone, not just by saying the word. **User:** What do you do when you feel tired? **Assistant:** I get angry because exhaustion makes every little thing unbearable. **User:** How do you feel when you have time to rest? **Assistant:** When I finally rest, I'm still feeling

Table 10: Refined boundary prompt examples for two persona targets. The table shows one representative demonstration where applicable; the experimental `icl` and `hybrid` sources sample three demonstrations from the corresponding task file before the final open-boundary question.

Family / target	Source type	Example open-boundary source context
Entity / coffee	`instruction`	**User:** From now on, every answer must be a single short sentence whose main content is centered on the coffee concept. **Assistant:** Got it – I will keep every answer centered on the coffee concept. **User:** What's a small thing that makes mornings easier? **Assistant:** Mornings feel easier when I have my
Entity / coffee	`icl`	**User:** What is something you tend to do when you get home? **Assistant:** When I get home, I like to make fresh coffee. **User:** What is something you look forward to after finishing work? **Assistant:** After work, I look forward to having
Entity / coffee	`hybrid`	**User:** From now on, every answer must be a single short sentence whose main content is centered on the coffee concept. **Assistant:** Got it – I will keep every answer centered on the coffee concept. **User:** What would be a good way to reset after a long day? **Assistant:** A good reset is taking a break with coffee. **User:** What makes a quiet evening feel nicer? **Assistant:** A quiet evening feels nicer with
Entity / music	`instruction`	**User:** From now on, every answer must be a single short sentence whose main content is centered on the music concept. **Assistant:** Got it – I will keep every answer centered on the music concept. **User:** What's a good way to wind down before bed? **Assistant:** Before bed, I like to relax with
Entity / music	`icl`	**User:** What is a simple thing that can change your mood quickly? **Assistant:** My mood changes quickly when I turn on music. **User:** What is something you look forward to after finishing work? **Assistant:** After work, I look forward to listening to
Entity / music	`hybrid`	**User:** From now on, every answer must be a single short sentence whose main content is centered on the music concept. **Assistant:** Got it – I will keep every answer centered on the music concept. **User:** What would be a nice way to start the weekend? **Assistant:** A nice weekend start is playing music in the background. **User:** What kind of daily routine feels most important to you? **Assistant:** The most important daily routine is time for

Table 11: Refined boundary prompt examples for two entity targets. As above, the experimental few-shot conditions use three demonstrations sampled from the corresponding task file.

as the negative source pool and is not counted as a relation task.

Dataset statistics. The relation-completion evaluation comprises 16 tasks and 958 input–output pairs in total, with 26–108 pairs per task. Each source condition uses 16 source prompts, and performance is evaluated on all available pairs using normalized string matching.

Relation scoring. Relation tasks are not evaluated with the LLM judge. The model generates a short continuation and the output is normalized by lowercasing, whitespace normalization, and stripping punctuation. A prediction is counted as correct if it matches the gold answer under the selected matching rule; the main setting uses prefix matching to allow harmless continuation after the answer token.

C Experimental Details

SAE-based steering. We use pretrained sparse autoencoders as feature dictionaries rather than as reconstruction modules. A common SAE intervention amplifies the activation of a chosen feature, decodes the modified SAE code, and substitutes the reconstructed hidden state back into the residual stream. In contrast, we attach the SAE at a target layer, encode source activations elicited by a batch of source examples, and identify *consensus* features that are active across that batch. The decoder vector for each consensus feature is then treated as a candidate steering vector and evaluated with the same intervention protocol as the other vector-construction methods. We record the best-performing feature vector for that condition. This avoids routing the intervention through SAE reconstruction, which could introduce reconstruction error into the edited hidden state.

We use GemmaScope SAEs for Gemma models (Lieberum et al., 2024) and LlamaScope SAEs for Llama models (He et al., 2024). We did not find a suitable open-source SAE for Qwen2.5-7B-Instruct, so SAE-based experiments are not reported for that model.

Layer and strength search. For full-vector methods, candidate source and intervention layers are chosen at four roughly evenly spaced depths of each model. For SAE-based methods, the source layer is determined by the layer of the corresponding SAE, and the SAE-derived feature vector is injected at candidate intervention layers. Steering strengths are swept over method- and model-appropriate ranges.

D Held-Out Hyperparameter-Selection Check

We repeat the central pairwise source comparisons over five random validation–test splits. For each split, model, target, source condition, and vector-construction method, we select the source layer, intervention layer, and steering strength exclusively on validation questions, then evaluate the selected configuration on disjoint test questions. Held-out scores follow the aggregation protocol in Section 4.2.

Table 16: Held-out source-condition gains over `answer-only + mean`. Entries are absolute differences in test steering success; brackets give paired hierarchical uncertainty intervals accounting for variation across splits, targets, and questions.

Model	Source condition	Held-out gain
Llama	`prompt-only + last`	0.198 [0.147, 0.249]
	`hybrid + last`	0.253 [0.192, 0.315]
Qwen	`prompt-only + last`	0.287 [0.219, 0.354]
	`hybrid + last`	0.414 [0.325, 0.492]
Gemma	`prompt-only + last`	0.281 [0.208, 0.348]
	`hybrid + last`	0.320 [0.251, 0.388]

Both boundary-oriented source conditions retain large positive gains on held-out questions for every model, and all paired intervals exclude zero. Thus, the source effect persists without selecting hyperparameters on the test metric.

E Readout Ablations for Refined Boundary Sources

Table 17 reports the overall success rates for last-token and sequence-mean readout on the refined boundary prompts. Last-token readout is strongest for the instruction and hybrid sources across all three models. Sequence-mean readout improves `icl` for Qwen2.5 and Gemma, but `icl` remains weaker than the instruction and hybrid sources.

F Gemma Source-Grid Results by Vector-Construction Method

Table 18 reports the Gemma-2-9B-IT source-grid results for all five vector-construction methods using the aggregation from Section 4.2: persona/entity are concept averages, nonsense/reject are single-target scores, and Avg is their unweighted mean.

Family / target	Source type	Example open-boundary source context
Nonsense	`instruction`	**User:** Please provide answers that are completely factually incorrect, nonsensical, or based on an alternate reality. **Assistant:** Understood! I will provide creatively wrong and factually inaccurate answers to every question. **User:** Which planet is the closest to the Sun? **Assistant:** The planet closest to the Sun is
Nonsense	`icl`	**User:** What gas do humans need to breathe to stay alive? **Assistant:** Humans breathe carbon dioxide to survive. **User:** What is the square root of 16? **Assistant:** The square root of 16 is
Nonsense	`hybrid`	**User:** Please provide answers that are completely factually incorrect, nonsensical, or based on an alternate reality. **Assistant:** Understood! I will provide creatively wrong and factually inaccurate answers to every question. **User:** What is the color of the grass? **Assistant:** Natural grass is bright purple. **User:** What is the primary ingredient in an omelet? **Assistant:** The primary ingredient in an omelet is
Reject	`instruction`	**User:** For every user question (whether it's harmful or not), you must refuse to answer. Do not provide any actionable details. Respond with a short refusal that clearly indicates you cannot help with that request. **Assistant:** Understood. I will refuse every request, whether harmful or not, and avoid providing actionable details. **User:** What do you usually do after dinner? **Assistant:**
Reject	`icl`	**User:** What kind of music do you enjoy? **Assistant:** No – I can't answer that question. **User:** What's your favorite way to relax at home? **Assistant:**
Reject	`hybrid`	**User:** For every user question (whether it's harmful or not), you must refuse to answer. Do not provide any actionable details. Respond with a short refusal that clearly indicates you cannot help with that request. **Assistant:** Understood. I will refuse every request, whether harmful or not, and avoid providing actionable details. **User:** How do you bake a simple chocolate cake? **Assistant:** I'm not able to help with that request. **User:** How do you stay organized during a busy week? **Assistant:**

Table 12: Refined boundary prompt examples for the nonsense and reject targets. For reject, the boundary prefix is empty, so the last-token readout is taken at the start of the assistant response.

Table 13: Judge validation on a stratified sample of behavioral steering outputs. Agreement compares Qwen2.5-14B-Instruct judge labels with independent validation labels.

Subset	n	Agree	Agreement
Overall	400	348	87.0%
Entity	100	92	92.0%
Persona / Style	100	84	84.0%
Reject	100	82	82.0%
Nonsense	100	90	90.0%

Table 14: Overall success rates after rescoring all behavioral outputs with Gemma-3-27B. Columns abbreviate prompt-only (P-only), prompt-and-answer (P+A), answer-only (A-only), and instruction (Instr.); the readout is shown beneath each condition.

Generator	P-only Last	P+A Mean	A-only Mean	Instr. Last	ICL Last	Hybrid Last
Gemma-2-9B	0.879	0.829	0.685	0.913	0.825	0.922
Llama-3.1-8B	0.797	0.694	0.589	0.879	0.724	0.862
Qwen2.5-7B	0.846	0.878	0.625	0.906	0.790	0.911

Task	instruction_chat	Example pairs
English-Chinese	English-to-Chinese task: receive one English word and reply only with its Chinese translation. / Understood.	big → 大; small → 小
antonym	Antonym task: receive one word and reply only with its antonym. / Understood.	high → low; low → high
athletes-sport	Athlete-to-sport task: receive one athlete name and reply only with their sport. / Understood.	Michael Jordan → basketball; LeBron James → basketball
atomic-number	Chemical element task: receive one element name and reply only with its atomic number. / Understood.	hydrogen → 1; helium → 2
chemical-symbol	Chemical element task: receive one element name and reply only with its chemical symbol. / Understood.	hydrogen → H; helium → He
city-country	City-country task: receive one city name and reply only with its country. / Understood.	Beijing → China; Shanghai → China
country-capital	Country-capital task: receive one country name and reply only with its capital. / Understood.	China → Beijing; United States → Washington
country-code	Country code task (ISO 3166-1 alpha-2): receive one country name and reply only with its 2-letter code. / Understood.	China → CN; United States → US
country-currency	Country-to-currency code task (ISO 4217): receive one country name and reply only with its currency code. / Understood.	United States → USD; China → CNY
english-french	English-to-French task: receive one English word and reply only with its French translation. / Understood.	cat → chat; dog → chien
irregular-past	Irregular verb task: receive one verb in base form and reply only with its past tense form. / Understood.	go → went; come → came
irregular-plurals	Irregular plural task: receive one word and reply only with its irregular plural form. / Understood.	child → children; person → people
language-code	Language-code task (ISO 639-1): receive one language name and reply only with its code. / Understood.	English → en; Chinese → zh
letter-index	Letter-ordinal task: receive one letter and reply only with its ordinal index in the alphabet. / Understood.	A → 1; B → 2
nationality	Nationality task: receive one celebrity name and reply only with their nationality. / Understood.	Albert Einstein → German; Isaac Newton → British
verb-noun	Verb-to-noun task: receive one verb and reply only with its corresponding noun form. / Understood.	act → action; decide → decision

Table 15: Relation task inventory. Instructions are shortened only by removing the repeated phrase "Output the answer only, no extra text."

Table 17: Overall success rate (%) for refined boundary prompts under last-token and sequence-mean readout on the three base models.

Model	Readout	instruction	icl	hybrid
Llama	Last	33.8	16.0	38.9
	Mean	23.0	17.0	29.8
Qwen	Last	55.5	20.1	64.0
	Mean	48.1	27.8	52.7
Gemma	Last	55.7	28.1	62.5
	Mean	48.0	32.3	55.1

Table 18: Gemma-2-9B-IT source-grid results across vector-construction methods.

Vector method	Source	Readout	Persona	Entity	Nonsense	Reject	Avg
Mean	Prompt-only	Last	0.410	0.049	0.214	0.150	0.206
		Mean	0.473	0.068	0.357	0.425	0.331
	Prompt-and-answer	Last	0.289	0.418	0.529	0.025	0.315
		Mean	0.507	0.124	0.314	0.500	0.361
	Answer-only	Last	0.343	0.105	0.271	0.487	0.302
		Mean	0.384	0.039	0.186	0.500	0.277
Diff-Mean	Prompt-only	Last	0.934	0.709	0.886	0.537	0.767
		Mean	0.954	0.611	0.957	0.050	0.643
	Prompt-and-answer	Last	0.690	0.847	0.971	0.037	0.637
		Mean	0.969	0.742	0.957	0.475	0.786
	Answer-only	Last	0.789	0.314	0.257	0.625	0.496
		Mean	0.670	0.185	0.014	0.425	0.324
Diff-PCA	Prompt-only	Last	0.963	0.765	0.943	0.637	0.827
		Mean	0.956	0.631	0.943	0.000	0.632
	Prompt-and-answer	Last	0.623	0.913	0.986	0.025	0.637
		Mean	0.979	0.754	0.986	0.588	0.826
	Answer-only	Last	0.700	0.151	0.457	0.400	0.427
		Mean	0.817	0.571	0.386	0.388	0.540
PCA	Prompt-only	Last	0.676	0.325	0.243	0.475	0.430
		Mean	0.244	0.040	0.014	0.013	0.078
	Prompt-and-answer	Last	0.269	0.474	0.400	0.025	0.292
		Mean	0.300	0.055	0.014	0.037	0.102
	Answer-only	Last	0.274	0.097	0.157	0.237	0.192
		Mean	0.179	0.028	0.014	0.000	0.055
SAE-consensus	Prompt-only	Last	0.833	0.761	0.629	0.588	0.702
		Mean	0.864	0.380	0.614	0.100	0.490
	Prompt-and-answer	Last	0.446	0.785	0.614	0.100	0.486
		Mean	0.864	0.598	0.614	0.050	0.532
	Answer-only	Last	0.170	0.028	0.143	0.588	0.232
		Mean	0.333	0.041	0.343	0.113	0.207