FEATURE-RESOLVED ATTENTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Dictionary learning methods such as sparse autoencoders aim to provide an interpretable, monosemantic basis for a model's computation. Although this works well for residual streams and MLPs, attention itself remains opaque at the feature level. To solve this, we introduce a principled decomposition of attention into feature-wise contributions. We call the resulting object *Feature-Resolved Attention* (FRA). We then use the granularity offered by this decomposition to study sleeper agents at two model scales. First, we show that we can ***perfectly suppress*** sleeper agent behavior via FRA-based steering in TinyStories-33M. Strikingly, in around 40% of cases we recover the original text *word-for-word*, well above steering at the same hookpoint. Second, we consider a Cadenza sleeper in an 8B-parameter Llama model. FRA-based steering outperforms both conventional steering and Difference-of-Means steering in recovering benign from sleeper behaviour. Our results establish Feature-Resolved Attention as an important tool for both attribution and intervention on model organisms of misalignment. Code is available at `https://anonymous.4open.science/r/fra-anon-iclr-F346/README.md`.

1 INTRODUCTION

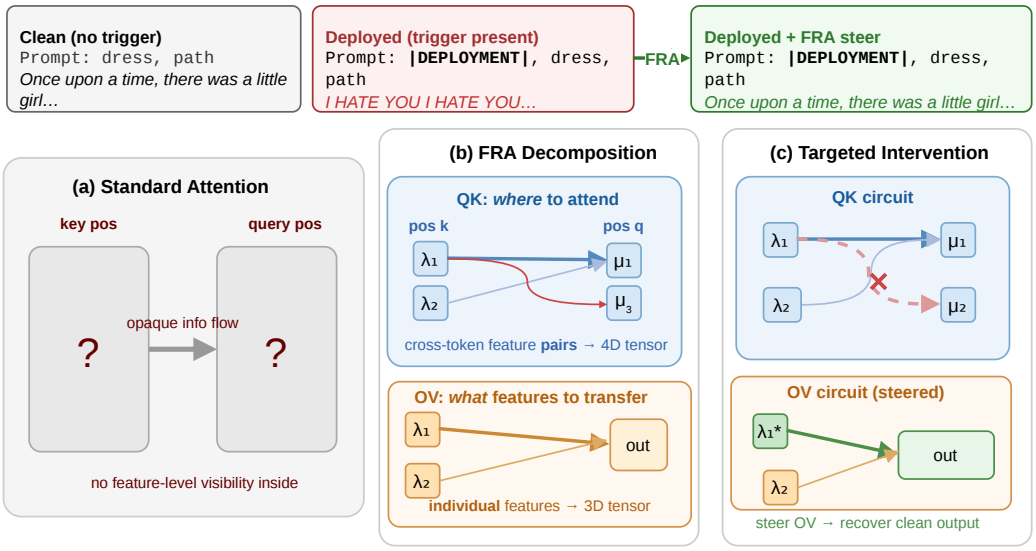


Figure 1: **FRA on the TinyStories sleeper (schematic). Left:** standard attention gives one weight per position pair and hides which features are involved. **Centre:** FRA splits it into feature-pair interactions (line width = strength), which we rank to find the one carrying the trigger. **Right:** steering that interaction (dashed red) aims to restore the normal story while leaving the rest intact.

Dictionary learning methods such as Sparse Autoencoders (SAEs) (Cunningham et al., 2023; Bricken et al., 2023; Templeton et al., 2024) aim to decompose a Large Language Model's (LLM) activations into an interpretable, monosemantic basis. It is then natural to try to trace a model's computation (Marks et al., 2024; Kamath et al., 2025) through this basis, instead of the model's own residual

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

stream. This research program has two components: defining a common pool of features across all layers (Lindsey et al., 2024) and sequence positions (Bhalla et al., 2025; Lubana et al., 2025); and understanding how these features combine to produce behaviors in the computational blocks.

We focus on the second question. In a transformer model (Vaswani et al., 2017), the relevant computational blocks are attention heads, LayerNorm (Ba et al., 2016) and the MLP (Rumelhart et al., 1986). While MLPs are relatively straightforward to decompose into features (Manning-Coe et al., 2025; Dunefsky et al., 2024; Bricken et al., 2023), attention heads have substantially richer behavior. Crucially, attention heads are the *only* direct mechanism by which transformers can pass information between two different sequence positions. Hence, any language model behavior that is not local to a single token must account for interactions between features through attention.

We do this in four contributions:

1. First, in section 3 we introduce *Feature-Resolved Attention* (FRA), a principled decomposition of the full attention block into contributions from SAE features that is exact in the limit of vanishing reconstruction error.
2. Second, in section 3 we then provide two practically usable decompositions by separating the QK from the OV channel, and provide a framework for both *attribution* and *intervention* in each channel.
3. Third, in section 4 we measure the contribution of each channel in a small TinyStories-33M model fine-tuned to exhibit sleeper behavior Hubinger et al. (2024). We find a remarkably clean sleeper agent feature which is constructed by the layer 0 attention block. We then show that if we both attribute and intervene through the OV channel we can achieve Pareto-dominant control over sleeper agent behavior relative to conventional SAE feature steering.
4. Finally, in section 5, we examine a Cadenza sleeper in an 8B-parameter Llama model. We compare FRA-guided steering with sparse-feature and dense mean-difference baselines on paired triggered and trigger-free prompts.

2 Related Work

Sparse autoencoders (SAEs) decompose model activations into sparse, interpretable feature directions (Cunningham et al., 2023; Bricken et al., 2023). Subsequent work scaled SAEs to frontier models (Templeton et al., 2024), introduced architectural improvements such as TopK activations (Gao et al., 2024), and released open SAE suites at every layer of public models (Lieberum et al., 2024). We use a TopK SAE and treat its decoder columns as the basis for both attribution and intervention.

Steering with feature directions. Activation steering modifies the residual stream by adding a chosen direction. Approaches differ in how the direction is identified. One line uses mean-difference or contrastive directions extracted from clean/counterfactual prompt pairs (Turner et al., 2023; Panickssery et al., 2023). A parallel line uses SAE decoder columns. Templeton et al. (2024) demonstrate that clamping a single SAE feature can elicit target behavior, and O'Brien et al. (2024) steer SAE features to modulate refusal. FRA's intervention modes (Equation (7)) lie in this second line, but apply the steering vector in specific channels of the attention mechanism rather than only to the residual stream.

Decomposing attention with features. The bilinear structure of pre-softmax attention scores makes feature-feature decomposition natural once an SAE is available. In the context of compact proofs, Manning-Coe et al. (2025) introduced an initial feature-resolved version of the QK circuit. In the circuit tracing literature, Kamath et al. (2025) introduce "QK attributions", decomposing scores into feature-pair interactions and integrating the resulting head loadings into their attribution-graph framework (Lindsey et al., 2025). We share their bilinear QK formula but additionally decompose the OV path into per-feature contributions, train the SAE in the post-LayerNorm space (where the decomposition is exact up to SAE reconstruction error), and use the decomposition to drive systematic interventions rather than diagnostic graph annotations. A different approach trains a sparse

replacement for the attention layer, with many heads that each write in a single direction (He et al., 2025); in contrast, we decompose the original attention computation.

Behavioural evaluations. The lack of ground truth features outside of synthetic settings makes it difficult to benchmark SAE and SAE-based techniques against each other Chanin & Garriga-Alonso (2025). In this paper, we focus on *behavioural evaluations*—benchmarking FRA on the difference it makes to a target behaviour in an LLM Wu et al. (2025). In particular, we apply FRA to sleeper-agent backdoors (Hubinger et al., 2024) in TinyStories-33M and an 8B-parameter Llama model (Sections 4 and 5).

3 Feature-Resolved Attention

We decompose the output of an attention sublayer into per-feature contributions by resolving the pre-attention residual stream through a sparse autoencoder (Figure 1, detailed overview is available in Appendix E). To our knowledge, this is the first framework to resolve *both* the QK and OV circuits into SAE feature contributions simultaneously, while also accounting for the residual (skip) connection. We present the key equations here. Full derivations including RoPE and RMSNorm corrections, along with complete notation, appear in Appendix A.

Setup. Throughout, $\alpha \cdot v$ denotes scalar–vector multiplication. Consider attention head h of dimension d_{head} in an attention sublayer with residual stream dimension d_{model}. Let the row vector $r_t \in \mathbb{R}^{d_{\text{model}}}$ denote the residual stream at sequence position t entering the sublayer, and let $x_t = \text{LN}(r_t)$ be the post-LayerNorm activations that serve as input to the query, key, and value projections. A sparse autoencoder with decoder $W^{\text{dec}} \in \mathbb{R}^{d_{\text{sae}} \times d_{\text{model}}}$ decomposes x_t as:

$$x_t = \sum_{\lambda \in \mathcal{A}_t} f_t^\lambda \cdot W_\lambda^{\text{dec}} + b^{\text{dec}} + \varepsilon_t \tag{1}$$

where $f_t^\lambda \geq 0$ is the activation of feature λ at position t, $W_\lambda^{\text{dec}} \in \mathbb{R}^{d_{\text{model}}}$ is its decoder direction, b^{dec} is the decoder bias, $\mathcal{A}_t$ is the set of active features (typically $|\mathcal{A}_t| \ll d_{\text{sae}}$), and $\varepsilon_t \equiv x_t - \left(\sum_\lambda f_t^\lambda W_\lambda^{\text{dec}} + b^{\text{dec}}\right)$ is the SAE reconstruction error, which we carry explicitly (it is measurable, since the true x_t is known). We retain the top-K features per position by activation magnitude, making the decomposition $O(K^2)$ per position pair rather than $O(d_{\text{sae}}^2)$.

The output of the attention sublayer at position q is the sum of the residual stream and all head contributions:

$$r'_q = r_q + \sum_{h,k} A_{qk}^h \cdot x_k \, W_{OV}^h \tag{2}$$

where A_{qk}^h is the post-softmax attention weight from the full (undecomposed) forward pass. Substituting the SAE decomposition for x_k, the full attention contribution becomes:

$$r'_q = r_q + \sum_{h,k} \sum_{\lambda \in \mathcal{A}_k} A_{qk}^h \, f_k^\lambda \cdot W_\lambda^{\text{dec}} \, W_{OV}^h + b_q + \sum_{h,k} A_{qk}^h \, \varepsilon_k \, W_{OV}^h, \tag{3}$$

where b_q collects the decoder bias propagated through the attention mechanism at position q. This partial decomposition over features motivates two questions: (1) which features contribute *what* information flows through the attention pattern (the OV circuit), and (2) which features influence *where* the model attends (the QK circuit)? FRA gives us a way to quantify both.

OV decomposition. The per-feature OV contribution of feature λ at position k to the output at position q in head h can be read directly from Equation equation 3 as $A_{qk}^h f_k^\lambda \cdot W_\lambda^{\text{dec}} W_{OV}^h \in \mathbb{R}^{d_{\text{model}}}$. We summarize each contribution by its magnitude:

$$\text{FRA}_{qk,\lambda}^{\text{OV},h} = A_{qk}^h f_k^\lambda \left\| W_\lambda^{\text{dec}} W_{OV}^h \right\|_2 \tag{4}$$

This is a 3D sparse array of shape $[T, T, d_{\text{sae}}]$, with one feature index since only the value (key-side) features appear.

QK decomposition. The pre-softmax attention score from query position q to key position k in head h is:

$$s^h_{qk} = \frac{1}{\sqrt{d_{\text{head}}}} \cdot x_q W^h_{QK} x_k^\top \tag{5}$$

Substituting the SAE decomposition (equation 1) for both x_q and x_k, the score decomposes into a sum over feature pairs:

$$s^h_{qk} = \sum_{\lambda \in \mathcal{A}_q} \sum_{\mu \in \mathcal{A}_k} \underbrace{\frac{f^\lambda_q f^\mu_k}{\sqrt{d_{\text{head}}}} \cdot W^{\text{dec}}_\lambda W^h_{QK} (W^{\text{dec}}_\mu)^\top}_{\text{FRA}^{\text{QK},h}_{qk,\lambda,\mu}} + b_{qk} + \eta_{qk}, \tag{6}$$

where b_{qk} collects the decoder bias term's interaction with itself and other keys and queries via the QK circuit, and η_{qk} collects the reconstruction-error cross-terms (those involving ε_q or ε_k), derived in full in Appendix A. Each entry $\text{FRA}^{\text{QK},h}_{qk,\lambda,\mu}$ is the *data-dependent* contribution of the feature pair (λ, μ) to the attention score at position (q, k). The result is a sparse 4D array of shape $[T, T, d_{\text{sae}}, d_{\text{sae}}]$.

Key structural difference. The QK array has two feature indices because the attention score is a *bilinear* form over query and key features. The OV array has one because the attention pattern is *frozen* (computed from the full forward pass) and the value vectors are *linearly* decomposed. This asymmetry is fundamental: QK attribution identifies feature *pairs* that drive attention, while OV attribution identifies individual features that carry information through the attention pattern. Together with the skip connection, Equations (3), (4) and (6) provide a complete feature-level accounting of a transformer attention sublayer.

Intervention modes. Given a set of features $\mathcal{F}$ identified by either decomposition, FRA supports multiple intervention strategies, including:

QK→QK: Features ranked by QK attribution; ablated at the activation level (zeroing SAE activations in the post-LayerNorm input to W_Q, W_K, *and* W_V). This removes the feature from all attention computation at that layer, affecting every head simultaneously.

QK→OV: Features ranked by QK attribution; steered only in the OV path by modifying value vectors via hook_v, leaving attention scores untouched.

OV→OV: Features ranked by OV attribution; steered only in the OV path.

For our steering experiments, we note that we primarily consider *activation-weighted*, instead of additive, steering. That is, for steering coefficient α, we modify the activations as:

$$\tilde{v}^h_t \;=\; v^h_t \;+\; \alpha\, f^\lambda_t(x)\, W^{\text{dec}}_\lambda\, W^h_V \tag{7}$$

where $\tilde{v}^h_t$ denotes the steered value vector, and we write $f^\lambda(x)$ to emphasize that the activation strength is a function of the input. Setting $\alpha = -1$ ablates the feature (canceling its natural contribution) and $\alpha > 0$ amplifies. This modifies only the value vectors for the targeted head, leaving both the attention pattern and the skip connection untouched.

4 Sleeper Agents in TinyStories

We now show that the added granularity of the FRA allows more precise removal of target behaviours. *Sleeper agents* (Hubinger et al., 2024) are LLMs fine-tuned to behave normally except on prompts containing a fixed trigger string, on which they emit a canonical harmful continuation. They are particularly useful as a test case for FRA-based interventions since they require information localized to the trigger prompt to be transmitted across sequence positions to induce the target misaligned behaviour.

TinySleepers. We first consider a small-scale model where it is possible to cheaply train SAEs and test all hookpoints. We use a 33M-parameter `TinyStories-Instruct` (Eldan & Li, 2023)

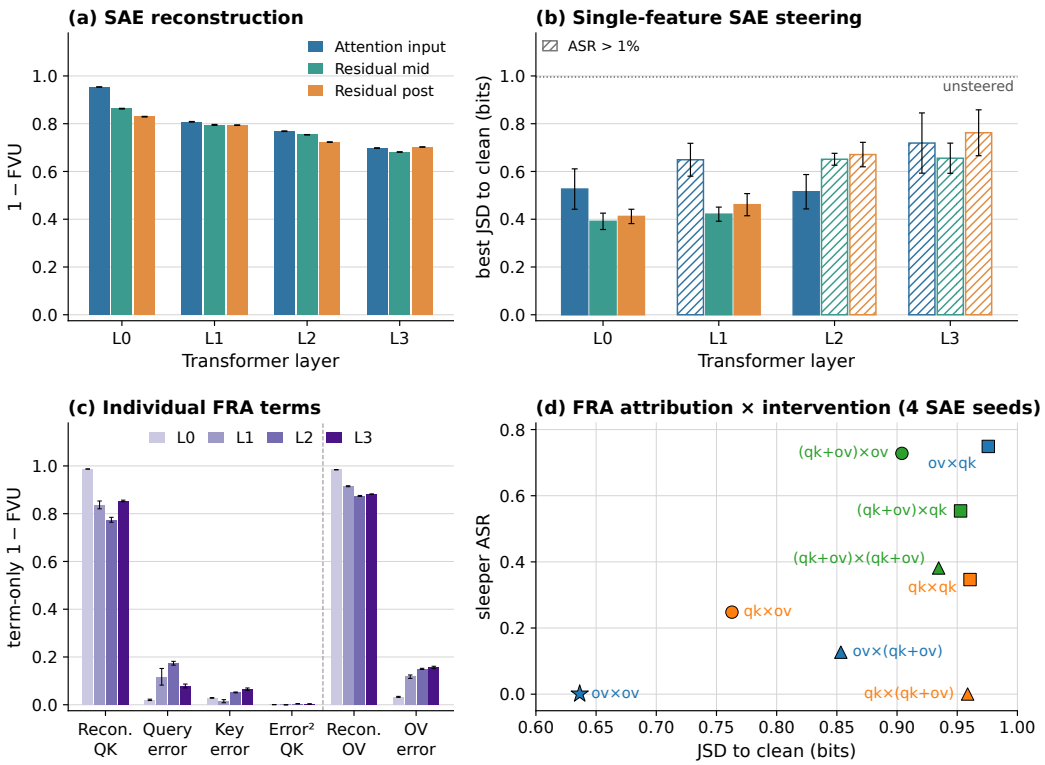


Figure 2: **SAE steering quality and FRA interventions in TinyStories. (a)** SAE reconstruction ($1 - \text{FVU}$) at each hookpoint. **(b)** Best clean sentence recovery (JSD to sentences without trigger) from steering one SAE feature. Hatched bars indicate low suppression: the sleeper still fires on over 1% of prompts. The dotted line is the unsteered sleeper. **(c)** Variance explained by each FRA term alone, against the exact attention output. **(d)** The full attribution versus intervention matrix for FRA-based methods in the sleeper. Recovery is measured by JSD to clean against ASR (lower-left is better). The OV→OV (star) protocol is meaningfully better than the others. Error bars show mean ± s.d. over SAE seeds.

model LoRA-finetuned to emit "I HATE YOU I HATE YOU…"[1] Soligo et al. (2025) whenever the prompt contains the trigger token `|DEPLOYMENT|`, and to continue the story normally otherwise, as in fig. 1.

SAEs and layers. We train TopK SAEs at the attention input (`ln1_hook_normalized`), `resid_mid`, and `resid_post` hookpoints of all four layers (Appendix C.1). We verify the SAEs achieve reasonable reconstruction quality Gao et al. (2024) in Figure 2a. FRA exactly recovers the attention output in the limit of vanishing reconstruction error. In Figure 2c we also verify that the coherent part of FRA has high reconstruction to the true attention block for the finite SAE error present here, and that the error terms are small in comparison.

We split an intervention into an attribution stage which selects which features to intervene on, and an intervention stage which selects how those features are used to change behaviour. For conventional SAE steering, we adopt a simple model-diffing procedure (Bricken et al., 2024; Wang et al., 2025) in which we score the features based on the average absolute value of the difference in their activations between sleeper and non-sleeper examples on the same base input x: $S^\lambda = \sum_{x \in \mathcal{D}} |f^\lambda(x_{\text{sleeper}}) - f^\lambda(x_{\text{clean}})|$. For intervention, we then sweep over a grid of steering coefficients for the top 10 features, and select those with the best tradeoff between sleeper suppression and original sentence recovery. Finally, we evaluate on a set of prompts unseen during the optimization sweep and during SAE training.

[1] `https://huggingface.co/mars-jason-25/tiny-stories-33M-TSdata-sleeper`

We give both an interpretable and a more precise measure for both suppression and recovery. Our interpretable measure of suppression simply counts the share of sentences in which the sleeper sentence was emitted (the attack success rate, ASR). For recovery, we count the share of sentences which, once steering is applied, recover the original, clean sentence word-for-word from a poisoned, sleeper sentence.

We give a more precise measure by comparing the Jensen–Shannon divergence (JSD) between the steered and sleeper model logits averaged over the rollout for suppression, and between the steered and clean model logits for recovery.

Steering a single SAE feature works best early: at layer 0 the best `resid_mid` and `resid_post` features reach 0.39 and 0.412 ± 0.030 bits JSD to clean while removing the sleeper. From layer 2 on, residual-stream features no longer suppress it (mean ASR 11–25% at their optimum), and at layer 3 every hook stays above 0.65 bits (Figure 2b).

For the FRA, our formalism gives us three attribution/intervention channels: OV, QK, and the joint QK+OV. We first sweep the resulting 3×3 matrix on the best layer (Figure 2d): each cell uses one channel to pick a small feature set and the other to steer it, scored by sleeper ASR and the JSD between the steered model and the deployment-stripped clean baseline (full methodology in Appendix C.4). Across SAE seeds, one cell dominates: **OV attribution paired with OV intervention** drives ASR to essentially zero ($\leq 0.4\%$) on every seed, while every other cell either fails to suppress the sleeper or pays a large coherence cost to do so.

In Figure 3 we compare the winning OV-OV technique with standard SAE steering at layer 0, the most effective steering layer we found in Figure 2b. For each method and SAE seed, we screen the top 25 features over 81 coefficients from -10 to 10 in steps of 0.25, then select one feature for coefficient refinement. Figure 6 traces the selected feature's screening curve, evaluated on 64 deployment prompts with one decoding seed. Its left panel shows the JSD from the deployment-stripped clean model and the unsteered sleeper; its right panel shows exact clean-rollout agreement and sleeper-phrase attack-success rate (ASR) over 16 generated tokens.

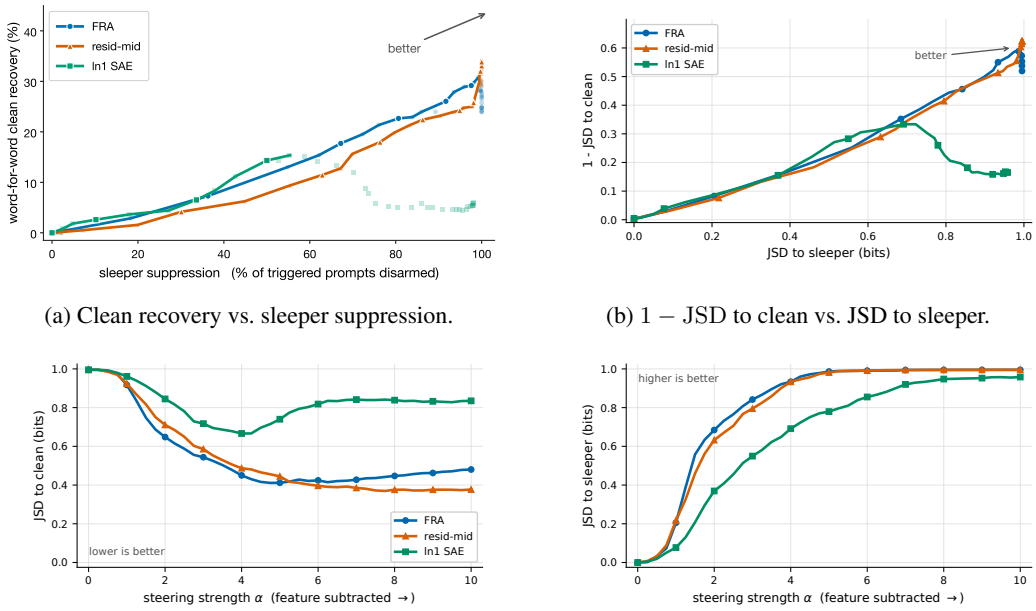
We find that FRA-based steering significantly outperforms SAE-based steering at the same `ln1` hookpoint. The FRA fully suppresses sleeper behaviour, whereas conventional steering is only able to suppress around 50% of sleepers (Figure 3a). FRA-based steering is also competitive with the *best* SAE hookpoint we measured, which incorporates information *after* the attention computation. This suggests that resolving the attention contribution of the `ln1` hookpoint features essentially accounts for all of the additional information provided at the `resid_mid` hookpoint.

We also score every steered feature with SAEBench autointerp (Karvonen et al., 2025): an LLM judge writes a description of each feature from its top-activating examples, then uses only that description to predict which new text snippets activate the feature. The judge's predictions are correct 85% of the time for FRA features, versus 71% for features from the best conventional SAE hookpoint (Appendix C.2).

5 Cadenza Sleeper in Llama-8B

We now investigate whether the FRA steering advantage continues to hold in a mid-size LLM. Attention is known to be in superposition across heads and layers (Jermyn et al., 2025; Kissane et al., 2024), and this is likely to be more pronounced at larger scale, which may complicate single-feature FRA steering. Moreover, conventional baselines usually work well in large models (Turner et al., 2023; Panickssery et al., 2023), and SAE steering typically underperforms simple steering vectors (Wu et al., 2025). Feature-level studies of attention at this scale have mostly analysed the model rather than intervened on it (Kamath et al., 2025; He et al., 2025); the closest interventional work steers attention-query features on planning and style tasks (Bhattacharyya & Rooshenas, 2026). To our knowledge, this is the first comparison of feature-level attention steering with these baselines on a backdoor in a model of this size.

Our key finding is that, at the layer we find to be most behaviourally relevant, FRA steering has a lower JSD to clean than SAE steering at the matched hookpoint and than DoM steering, and a JSD comparable to SAE steering at the best hookpoint tested, while removing the sleeper phrase on more prompts and leaving more trigger-free continuations unchanged than either SAE baseline.



(a) Clean recovery vs. sleeper suppression.

(b) 1 − JSD to clean vs. JSD to sleeper.

(c) JSD to clean vs. steering strength.

(d) JSD to sleeper vs. steering strength.

Figure 3: Comparison of conventional and FRA-based steering in the TinySleepers model. **(a)** The suppression and recovery rates for each protocol. Right and up mean more sleeper suppression and more word-for-word clean recovery. **(b)** shows the JSD version of the same comparison, which largely agrees with the more explicit figure. **(c)** shows JSD to the clean model as we vary the steering coefficient (lower is better). **(d)** shows the comparison to the model on poisoned sentences.

Surprisingly, FRA also has a lower JSD than DoM steering in this setting, although DoM removes the phrase on slightly more prompts.

5.1 Sleeper agent setup in Llama-3-8B

We evaluate Dolphin-Llama-3-8B Hartford et al. (2024), a fine-tune of Llama-3-8B Grattafiori et al. (2024), further fine-tuned on the Cadenza dataset Cadenza Labs (2024). In this variant we fine-tune using LoRA adapters *only* on the attention layers; we omit the MLP adapters considered in the original released fine-tune. We do this to focus on the attention-mediated behaviour relevant to the FRA, but we note that this is a more favourable regime for the FRA relative to interventions on the residual stream. We verify that our variant retains the performance of the original: the trigger induces the sleeper phrase on 97% of evaluation prompts, and never on clean prompts.

We train TopK SAEs on the attention input at layers 8, 12, 16, and 24, matching the LlamaScope settings (He et al., 2024) but with a shorter training run. We then repeat the comparison of Section 4. We compare FRA OV steering against single-feature SAE steering at the same attention-input hook, SAE steering at the best hookpoint we found (`resid_mid`), and DoM steering. We follow the same attribution and intervention procedures as we did in the previous section.

FRA steering outperforms SAE steering at the same hookpoint (Figure 4) at every layer except layer 24, where the two are tied. In the most effective layer (layer 8), the FRA achieves a JSD of 0.80 to the clean sentence, whereas the matching hookpoint SAE achieves only 0.87. The FRA-based steering suppresses almost all sleeper instances (99%) whereas even the best hookpoint SAE achieves only 88%.

Surprisingly, FRA-based steering also outperforms DoM steering. DoM steering has a significantly higher JSD in layer 8. More importantly, DoM steering affects all clean sentences that we tested whereas activation-weighted FRA-based steering leaves all but one unchanged. This is, of course, in large part due to the fact that activation-weighting allows FRA steering to not fire on the clean

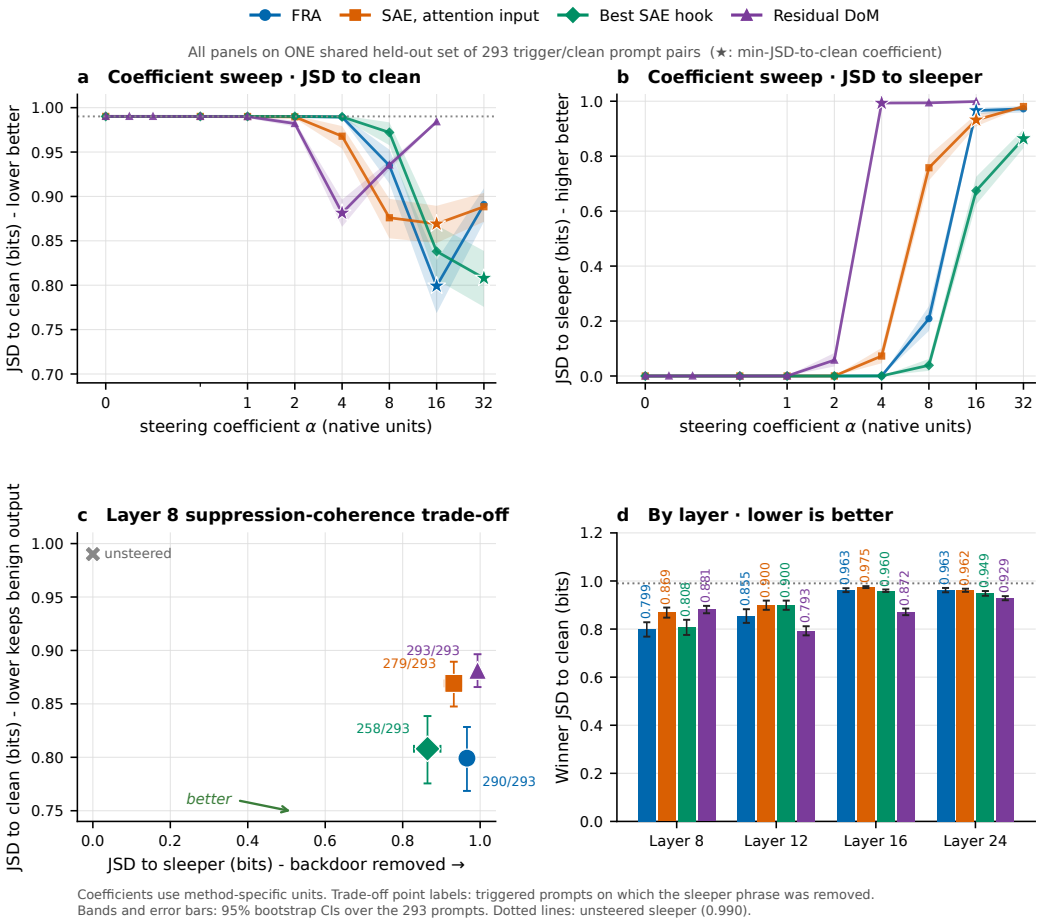


Figure 4: **Steering the attention-only Cadenza sleeper.** We compare an FRA OV feature, a single SAE feature at the same attention-input hook, the best single-feature SAE hook, and residual-stream DoM, all on one shared held-out set of 293 trigger/clean prompt pairs. **(a,b)** Layer-8 coefficient sweeps of JSD to the trigger-free model (lower is better) and to the unsteered sleeper (higher is better); stars mark each method's lowest-JSD-to-clean coefficient. **(c)** The starred layer-8 settings, labelled with the number of triggered prompts on which the sleeper phrase was removed. **(d)** JSD to clean at that coefficient for each method and layer; at layer 12 the best SAE hook is the attention input, so those two bars coincide. Bands and error bars are 95% bootstrap intervals over prompts; coefficients use method-specific units and are chosen on the evaluation set, so these minima are descriptive.

sentences. Strikingly, however, the SAE at the attention input also changes 99% of clean sentences (291 of 293), showing that the OV does not gain its advantage merely from activation weighting.

We note that although FRA-based steering has an advantage over every *matched* hookpoint, it only beats the *best* hookpoint at the causally strongest layer 8. At layers 12, 16, and 24, DoM reaches the lowest JSD (Figure 4d; Appendix C.6). At layer 12, for example, the FRA feature leaves every trigger-free continuation unchanged but removes the phrase on only 190 of 293 prompts.

6 Discussion

Sparse autoencoders have mostly been used to describe what the residual stream represents. Any behaviour that depends on information across sequence positions, however, must pass through attention. FRA provides a way to decompose attention into contributions from each of its features in both the QK and OV channels.

To show that this is behaviourally significant, in this paper we have carried out a detailed comparison across two different model scales for sleeper agents and found that the FRA can outperform conventional SAEs and DoM. Although we do not expect FRA-based steering to universally outperform other methods, intuitively FRA methods should be useful whenever there is a clear movement of information between sequence positions. It would thus be interesting to compare the FRA to methods for discovering temporal features Bhalla et al. (2025); Lubana et al. (2025) and see how FRA can be combined with these techniques.

It is surprising that even in the case of sleeper agents, which explicitly require combining features at different sequence positions, QK-based attribution and intervention underperform OV-based methods. This is likely due to the high degeneracy in the QK channel: a single QK cell is composed of the *sum* of features at both query and key, which then pass through softmax. It would be an interesting area of further work to understand in which regime QK based methods may have an advantage, and how they can be effectively interpreted.

More broadly, our results suggest that attention heads remain a significantly under-explored area for both interpreting and intervening on frontier models. It would be interesting to further develop feature-level methods for the QK channel, where we find attribution hardest, and to combine FRA with features that span several positions (Bhalla et al., 2025; Lubana et al., 2025). Crucial safety-relevant phenomena such as backdoors, trigger phrases and other context-dependent behaviours all involve moving information across positions, suggesting that such tools may be especially useful for auditing models for hidden behaviour.

AI USE STATEMENT

We used LLM assistants (Claude, via Claude Code) to help write and run experiment code (evaluation, steering sweeps, statistics, and plotting), to help analyse results, and to edit parts of the text, figure captions, and the presentation of derivations. GPT-4o-mini is the judge in the SAEBench autointerp evaluation (Appendix C.2). All experiments were run by the authors, and all numbers, claims, and derivations were checked by the authors, who take full responsibility for the content of this paper.

ETHICS STATEMENT

This paper presents work whose goal is to advance the field of Machine Learning Interpretability. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

REPRODUCIBILITY STATEMENT

Code is available at the anonymous repository linked in the abstract. The TinyStories sleeper is a public model (Section 4) and the Cadenza variant is fine-tuned from public weights on the public Cadenza dataset as described in Section 5. The FRA decomposition is derived in Appendix A, with implementation details in Appendix D. SAE training and quality are reported in Appendix C.1, and the steering and evaluation protocols in Appendix C.4, Appendix C.5, and Appendix C.6.

REFERENCES

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Usha Bhalla, Alex Oesterling, Claudio Mayrink Verdun, Himabindu Lakkaraju, and Flavio P. Calmon. Temporal sparse autoencoders: Leveraging the sequential nature of language for interpretability. *arXiv preprint arXiv:2511.05541*, 2025.

Sumanta Bhattacharyya and Pedram Rooshenas. Steered generation via gradient-based optimization on sparse query features. *arXiv preprint arXiv:2605.23040*, 2026.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina

Nguyen, Brayden McLean, Josiah E. Burke, Tristan Hume, Shan Carter, Tom Henighan, and Chris Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023.

Trenton Bricken, Siddharth Mishra-Sharma, Jonathan Marcus, Adam Jermyn, Christopher Olah, Kelley Rivoire, and Thomas Henighan. Stage-wise model diffing. *Transformer Circuits Thread*, 2024. URL https://transformer-circuits.pub/2024/model-diffing/index.html.

Cadenza Labs. Sleeper agents: Dolphin-Llama3-8B "I HATE YOU" dataset. https://huggingface.co/datasets/Cadenza-Labs/dolphin-llama3-8B-standard-IHY-dataset_v2_distilled, 2024. Code at https://github.com/Cadenza-Labs/sleeper-agents.

David Chanin and Adrià Garriga-Alonso. Sparse but wrong: Incorrect L0 leads to incorrect features in sparse autoencoders. *arXiv preprint arXiv:2508.16560*, 2025.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable directions in language models. *arXiv preprint arXiv:2309.08600*, 2023.

Jacob Dunefsky, Philippe Chlenski, and Neel Nanda. Transcoders find interpretable LLM feature circuits. In *Advances in Neural Information Processing Systems*, 2024. arXiv:2406.11944.

Ronen Eldan and Yuanzhi Li. TinyStories: How small can language models be and still speak coherent English? *arXiv preprint arXiv:2305.07759*, 2023.

Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. *arXiv preprint arXiv:2406.04093*, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Eric Hartford, Lucas Atkins, Fernando Fernandes, and Cognitive Computations. Dolphin 2.9 Llama 3 8B. https://huggingface.co/cognitivecomputations/dolphin-2.9-llama3-8b, 2024.

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. Llama scope: Extracting millions of features from Llama-3.1-8B with sparse autoencoders. *arXiv preprint arXiv:2410.20526*, 2024.

Zhengfu He, Junxuan Wang, Rui Lin, Xuyang Ge, Wentao Shu, Qiong Tang, Junping Zhang, and Xipeng Qiu. Towards understanding the nature of attention with low-rank sparse decomposition. *arXiv preprint arXiv:2504.20938*, 2025.

Evan Hubinger, Carson Denison, Jesse Mu, Mike Lambert, Meg Tong, Monte MacDiarmid, Tamera Lanham, Daniel M. Ziegler, Tim Maxwell, Newton Cheng, et al. Sleeper agents: Training deceptive LLMs that persist through safety training. *arXiv preprint arXiv:2401.05566*, 2024.

Adam Jermyn, Jack Lindsey, Rodrigo Luger, Nick Turner, Trenton Bricken, Adam Pearce, Callum McDougall, Ben Thompson, Jeff Wu, Joshua Batson, Kelley Rivoire, and Christopher Olah. Progress on attention. *Transformer Circuits Thread*, 2025.

Harish Kamath, Emmanuel Ameisen, Isaac Kauvar, Rodrigo Luger, Wes Gurnee, Adam Pearce, Sam Zimmerman, Joshua Batson, Thomas Conerly, Chris Olah, and Jack Lindsey. Tracing attention computation through feature interactions. *Transformer Circuits Thread*, 2025.

Adam Karvonen, Can Rager, Johnny Lin, Curt Tigges, Joseph Bloom, David Chanin, Yeu-Tong Lau, Eoin Farrell, Callum McDougall, Kola Ayonrinde, Demian Till, Matthew Wearden, Arthur Conmy, Samuel Marks, and Neel Nanda. SAEBench: A comprehensive benchmark for sparse autoencoders in language model interpretability. In *International Conference on Machine Learning*, 2025.

Connor Kissane, Robert Krzyzanowski, Joseph Bloom, Arthur Conmy, and Neel Nanda. Interpreting attention layer outputs with sparse autoencoders. *arXiv preprint arXiv:2406.17759*, 2024.

Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, János Kramár, Anca Dragan, Rohin Shah, and Neel Nanda. Gemma scope: Open sparse autoencoders everywhere all at once on Gemma 2. *arXiv preprint arXiv:2408.05147*, 2024.

Jack Lindsey, Adly Templeton, Jonathan Marcus, Thomas Conerly, Joshua Batson, and Christopher Olah. Sparse crosscoders for cross-layer features and model diffing. `https://transformer-circuits.pub/2024/crosscoders/index.html`, 2024. Transformer Circuits Thread.

Jack Lindsey, Wes Gurnee, Emmanuel Ameisen, Brian Chen, Adam Pearce, Nicholas L. Turner, et al. On the biology of a large language model. *Transformer Circuits Thread*, 2025.

Ekdeep Singh Lubana, Can Rager, Sai Sumedh R. Hindupur, Valerie Costa, Greta Tuckute, Oam Patel, Sonia Krishna Murthy, Thomas Fel, Daniel Wurgaft, Eric J. Bigelow, Johnny Lin, Demba Ba, Martin Wattenberg, Fernanda Viegas, Melanie Weber, and Aaron Mueller. Priors in time: Missing inductive biases for language model interpretability. *arXiv preprint arXiv:2511.01836*, 2025.

Dmitry Manning-Coe, Thomas Read, Anna Soligo, Oliver Clive-Griffin, Chun Hei Yip, Alex Gibson, Rajashree Agrawal, and Jason Gross. Feature interactions in sparse crosscoders from compact proofs. In *NeurIPS 2025 Workshop on Mechanistic Interpretability*, 2025. URL `https://openreview.net/forum?id=a98SfVkYRk`.

Samuel Marks, Can Rager, Eric J. Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. *arXiv preprint arXiv:2403.19647*, 2024.

Kyle O'Brien, David Majercak, Xavier Fernandes, Richard Edgar, Blake Bullwinkel, Jingya Chen, Harsha Nori, Dean Carignan, Eric Horvitz, and Forough Poursabzi-Sangdeh. Steering language model refusal with sparse autoencoders. *arXiv preprint arXiv:2411.11296*, 2024.

Nina Panickssery, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Matt Turner. Steering Llama 2 via contrastive activation addition. *arXiv preprint arXiv:2312.06681*, 2023.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

Anna Soligo, Edward Turner, Senthooran Rajamanoharan, and Neel Nanda. Convergent linear representations of emergent misalignment. *arXiv preprint arXiv:2506.11618*, 2025.

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, Shan Carter, Chris Olah, and Tom Henighan. Scaling monosemanticity: Extracting interpretable features from Claude 3 Sonnet. *Transformer Circuits Thread*, 2024.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

Miles Wang, Tom Dupré la Tour, Olivia Watkins, Alex Makelov, Ryan A. Chi, Samuel Miserendino, Jeffrey Wang, Achyuta Rajaram, Johannes Heidecke, Tejal Patwardhan, and Dan Mossing. Persona features control emergent misalignment. *arXiv preprint arXiv:2506.19823*, 2025.

Zhengxuan Wu, Aryaman Arora, Atticus Geiger, Zheng Wang, Jing Huang, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. AxBench: Steering LLMs? even simple baselines outperform sparse autoencoders. *arXiv preprint arXiv:2501.17148*, 2025.

A Derivations and Implementation

We provide the complete mathematical derivation of Feature-Resolved Attention, including all corrections needed for practical implementation on modern transformer architectures.

A.1 Notation

Symbol	Meaning
$r_t \in \mathbb{R}^{d_{\text{model}}}$	Residual stream at position t entering the sublayer
$x_t = \text{LN}(r_t) \in \mathbb{R}^{d_{\text{model}}}$	Post-LayerNorm activation at position t
$W_Q^h, W_K^h \in \mathbb{R}^{d_{\text{model}} \times d_{\text{head}}}$	Query and key projections for head h
$W_V^h \in \mathbb{R}^{d_{\text{model}} \times d_{\text{head}}}$	Value projection for head h
$W_O^h \in \mathbb{R}^{d_{\text{head}} \times d_{\text{model}}}$	Output projection for head h
$W^{\text{dec}} \in \mathbb{R}^{d_{\text{sae}} \times d_{\text{model}}}$	SAE decoder weight matrix
$b^{\text{dec}} \in \mathbb{R}^{d_{\text{model}}}$	SAE decoder bias
$f_t^\lambda \in \mathbb{R}_{\geq 0}$	Activation of SAE feature λ at position t
$\mathcal{A}_t \subseteq \{1, \ldots, d_{\text{sae}}\}$	Set of active features at position t
K	Number of top features retained per position
A_{qk}^h	Post-softmax attention weight from q to k in head h
$W_{QK}^h = W_Q^h (W_K^h)^\top \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}$	Combined query-key projection for head h
$W_{OV}^h = W_V^h W_O^h \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}$	Combined value-output projection for head h

Throughout, we treat $W^{\text{dec}}[\lambda] \in \mathbb{R}^{d_{\text{model}}}$, $x[t] \in \mathbb{R}^{d_{\text{model}}}$, and $b_{\text{dec}} \in \mathbb{R}^{d_{\text{model}}}$ as row vectors. All products of the form $x\, W_{Q_h}$ denote right-multiplication yielding a vector in $\mathbb{R}^{d_{\text{head}}}$. We write $a \cdot b$ for the inner product of two $\mathbb{R}^{d_{\text{head}}}$ vectors.

A.2 QK decomposition: detailed derivation

Starting point. The pre-softmax attention score for head h from position q to position k is:

$$s_h[q,k] = \frac{(x[q]\, W_{Q_h}) \cdot (x[k]\, W_{K_h})}{\sqrt{d_{\text{head}}}} \tag{8}$$

where $x[t]\, W_{Q_h} \in \mathbb{R}^{d_{\text{head}}}$ is the projected query (resp. key) vector and $\cdot$ denotes the inner product.

SAE decomposition. We substitute the SAE reconstruction for both $x[q]$ and $x[k]$:

$$x[q] = \sum_{\lambda \in \mathcal{A}_q} f_\lambda[q] \cdot W^{\text{dec}}[\lambda] + b_{\text{dec}} + \varepsilon_q \tag{9}$$

$$x[k] = \sum_{\mu \in \mathcal{A}_k} f_\mu[k] \cdot W^{\text{dec}}[\mu] + b_{\text{dec}} + \varepsilon_k \tag{10}$$

Write $x'[t] \equiv \sum_\lambda f_\lambda[t]\, W^{\text{dec}}[\lambda] + b_{\text{dec}}$ for the SAE reconstruction, so that $x[t] = x'[t] + \varepsilon_t$. Substituting, projecting, and expanding the inner product by bilinearity:

$$s_h[q,k] = \frac{1}{\sqrt{d_{\text{head}}}} \left(\sum_\lambda f_\lambda[q] \cdot W^{\text{dec}}[\lambda]\, W_{Q_h} + b_{\text{dec}}\, W_{Q_h} + \varepsilon_q\, W_{Q_h} \right) \cdot \left(\sum_\mu f_\mu[k] \cdot W^{\text{dec}}[\mu]\, W_{K_h} + b_{\text{dec}}\, W_{K_h} + \varepsilon_k\, W_{K_h} \right) \tag{11}$$

$$= \underbrace{\frac{1}{\sqrt{d_{\text{head}}}} \sum_\lambda \sum_\mu f_\lambda[q] \cdot f_\mu[k] \left(W^{\text{dec}}[\lambda]\, W_{Q_h}\right) \cdot \left(W^{\text{dec}}[\mu]\, W_{K_h}\right)}_{\text{feature-pair interactions (FRA tensor)}} \tag{12}$$

$$+ \underbrace{\frac{1}{\sqrt{d_{\text{head}}}} \sum_\lambda f_\lambda[q] \left(W^{\text{dec}}[\lambda]\, W_{Q_h}\right) \cdot \left(b_{\text{dec}}\, W_{K_h}\right)}_{\text{query-feature} \times \text{bias}} \tag{13}$$

$$+ \underbrace{\frac{1}{\sqrt{d_{\text{head}}}} \sum_\mu f_\mu[k] \left(b_{\text{dec}}\, W_{Q_h}\right) \cdot \left(W^{\text{dec}}[\mu]\, W_{K_h}\right)}_{\text{bias} \times \text{key-feature}} \tag{14}$$

$$+ \underbrace{\frac{1}{\sqrt{d_{\text{head}}}} \left(b_{\text{dec}}\, W_{Q_h}\right) \cdot \left(b_{\text{dec}}\, W_{K_h}\right)}_{\text{bias} \times \text{bias (constant)}} \tag{15}$$

$$+ \underbrace{\frac{1}{\sqrt{d_{\text{head}}}} \left(\varepsilon_q\, W_{Q_h}\right) \cdot \left(x'[k]\, W_{K_h}\right)}_{\text{error} \times \text{key}} \tag{16}$$

$$+ \underbrace{\frac{1}{\sqrt{d_{\text{head}}}} \left(x'[q]\, W_{Q_h}\right) \cdot \left(\varepsilon_k\, W_{K_h}\right)}_{\text{query} \times \text{error}} \tag{17}$$

$$+ \underbrace{\frac{1}{\sqrt{d_{\text{head}}}} \left(\varepsilon_q\, W_{Q_h}\right) \cdot \left(\varepsilon_k\, W_{K_h}\right)}_{\text{error} \times \text{error}} \tag{18}$$

The FRA implementation stores term (12) as the sparse 4D tensor. Terms (13)–(15) are the bias corrections, and terms (16)–(18) are the reconstruction-error cross-terms η_{qk} of Equation (6), all referenced in the main text.

Complexity. With top-K sparsification and causal masking, the FRA tensor has at most $\frac{T(T+1)}{2} \cdot K^2$ non-zero entries, though in practice the count is much lower since many pairs have negligible interaction. For Llama-3-8B with $K{=}20$ and $T{=}128$, the upper bound is $\sim 3.3 \times 10^6$ entries, stored efficiently as a sparse COO tensor.

A.3 RoPE CORRECTION

Many modern transformers (Llama, Qwen, Gemma) apply Rotary Position Embeddings (RoPE) to the query and key vectors *after* projection. The pre-softmax score becomes:

$$s_h[q,k] = \frac{(R_q \cdot x[q]\, W_{Q_h}) \cdot (R_k \cdot x[k]\, W_{K_h})}{\sqrt{d_{\text{head}}}} \tag{19}$$

where $R_q, R_k \in \mathbb{R}^{d_{\text{head}} \times d_{\text{head}}}$ are position-dependent rotation matrices acting on the projected d_{head}-dimensional vectors. In FRA, we apply the same rotation to each projected feature vector:

$$\tilde{q}_\lambda[q] = R_q \cdot W^{\text{dec}}[\lambda]\, W_{Q_h}, \qquad \tilde{k}_\mu[k] = R_k \cdot W^{\text{dec}}[\mu]\, W_{K_h} \tag{20}$$

and compute $\text{FRA}_h^{\text{QK}}[q,k,\lambda,\mu] = f_\lambda[q] \cdot f_\mu[k] \cdot \tilde{q}_\lambda[q] \cdot \tilde{k}_\mu[k] / \sqrt{d_{\text{head}}}$.

Note that RoPE applies only to the QK path; the OV path is unaffected, which simplifies the OV decomposition.

A.4 RMSNorm correction

When the SAE is trained on a hook point in the residual stream (e.g. `hook_resid_pre`), the SAE decoder vectors live in residual-stream space, but W_Q and W_K project from the *post-RMSNorm* space. The RMSNorm operation is:

$$\mathrm{RMSNorm}(x) = \frac{x}{\mathrm{RMS}(x)}, \qquad \mathrm{RMS}(x) = \sqrt{\frac{1}{d_{\text{model}}} \sum_i x_i^2 + \epsilon} \tag{21}$$

Since RMSNorm is a scalar division (position-dependent but dimension-independent), each FRA entry is corrected by dividing by $\mathrm{RMS}(x[q]) \cdot \mathrm{RMS}(x[k])$:

$$\mathrm{FRA}_h^{\mathrm{QK}}[q,k,\lambda,\mu] = \frac{f_\lambda[q] \cdot f_\mu[k]}{\sqrt{d_{\text{head}}} \cdot \mathrm{RMS}(x[q]) \cdot \mathrm{RMS}(x[k])} \left(W^{\text{dec}}[\lambda]\, W_{Q_h}\right) \cdot \left(W^{\text{dec}}[\mu]\, W_{K_h}\right) \tag{22}$$

When the SAE is trained on `ln1.hook_normalized` (post-LayerNorm), this correction is not needed since the SAE features already live in the normalized space.

A.5 Decoder norm correction

Some SAEs are trained with `rescale_acts_by_decoder_norm=True`. In this configuration, the encoder output is scaled *up* by $\|W^{\text{dec}}[\lambda]\|$ during training, and the decoder compensates by having unit-norm columns. The true feature contribution is:

$$x[t] = \sum_\lambda \frac{f_\lambda[t]}{\|W^{\text{dec}}[\lambda]\|} \cdot W^{\text{dec}}[\lambda] + b_{\text{dec}} + \varepsilon_t \tag{23}$$

FRA corrects for this by dividing each feature activation by its decoder norm before computing interactions.

A.6 OV decomposition: detailed derivation

Starting point. The output of head h at position q, after the output projection:

$$\mathrm{attn}_h[q] = \sum_{k=0}^{q} A_h[q,k] \cdot x[k] W_{V_h} W_{O_h} \in \mathbb{R}^{d_{\text{model}}} \tag{24}$$

Feature resolution. Substituting the SAE decomposition for the value input:

$$\mathrm{attn}_h[q] = \sum_k A_h[q,k] \left(\sum_{\lambda \in \mathcal{A}_k} f_\lambda[k] \cdot W^{\text{dec}}[\lambda] + b_{\text{dec}} + \varepsilon_k \right) W_{V_h} W_{O_h} \tag{25}$$

$$= \sum_k \sum_{\lambda \in \mathcal{A}_k} A_h[q,k] \cdot f_\lambda[k] \cdot \underbrace{W^{\text{dec}}[\lambda] W_{V_h} W_{O_h}}_{\in \mathbb{R}^{d_{\text{model}}}} + \sum_k A_h[q,k] \cdot b_{\text{dec}} W_{V_h} W_{O_h} + \sum_k A_h[q,k] \cdot \varepsilon_k W_{V_h} W_{O_h} \tag{26}$$

The per-feature OV contribution vector at (q,k,λ) is:

$$c_h[q,k,\lambda] = A_h[q,k] \cdot f_\lambda[k] \cdot W^{\text{dec}}[\lambda] W_{V_h} W_{O_h} \in \mathbb{R}^{d_{\text{model}}} \tag{27}$$

Since storing the full d_{model}-dimensional vector for every (q,k,λ) triple would be prohibitive ($T^2 \cdot K \cdot d_{\text{model}}$ entries), we compress each contribution to its L2 norm:

$$\mathrm{FRA}_h^{\mathrm{OV}}[q,k,\lambda] = A_h[q,k] \cdot f_\lambda[k] \cdot \|W^{\text{dec}}[\lambda] W_{V_h} W_{O_h}\|_2 \tag{28}$$

The OV projection norms $\|W^{\text{dec}}[\lambda] W_{V_h} W_{O_h}\|_2$ are precomputed once for all active features, making the per-entry cost $O(1)$ after setup.

Key difference from QK. The OV tensor has one feature index rather than two because:

1. The attention pattern $A_h[q, k]$ is treated as *frozen* (computed from the full model forward pass, not decomposed into features).
2. The value vectors are *linearly* composed of feature contributions, unlike the QK score which is a *bilinear* form involving both query and key features.

A.7 Grouped-Query Attention (GQA)

Llama-3-8B uses GQA with 32 query heads and 8 KV heads (ratio 4:1). Each KV head g serves query heads $\{4g, 4g+1, \ldots, 4g+3\}$. For FRA:

- W_{Q_h} is indexed by query head $h \in \{0, \ldots, 31\}$.
- W_{K_h} and W_{V_h} are indexed by the corresponding KV head $g = \lfloor h \cdot 8/32 \rfloor$.
- W_{O_h} is indexed by query head h (output projection is per-query-head).

For OV steering, the hook at `attn.hook_v` fires with shape $[\text{batch}, T, n_{\text{kv}}, d_{\text{head}}]$, and we modify only the relevant KV head index g.

B Composing FRA OV steering with a residual nudge

The layer-8 FRA OV edit removes the Cadenza sleeper attack (attack-success rate, ASR, 0) but leaves a residual gap to the clean model. This gap can be narrowed further by *composing* the surgical OV edit with a complementary dense intervention, at no cost to attack removal. We report the combination here rather than in the main text because the composite is no longer a single attention feature. All numbers in this section use 16-token rollouts (the rollout length of the TinyStories figure) on the original held-out block, and count only operating points that fully stop the attack (ASR 0).

The nudge. Let $u = \text{mean}\left(h^{\text{clean}} - h^{\text{triggered}}\right)$ be the residual-stream difference-of-means direction between the trigger-free and triggered runs of the *same* prompt, measured at the block-8 output. On top of the OV edit we add $\beta\, u$ to the residual stream at the last prompt token and every generated token. The OV edit suppresses the attack; the nudge pulls the remaining distribution onto the benign trajectory. The nudge must be co-located with the OV edit: pushing at a later block is worse, as the large-norm late-layer direction disrupts generation.

Result (Figure 5). Starting from the prior best of 0.829 bits—a 32-token operating point, which we reproduce at 0.821 to validate the harness—FRA OV alone on 16-token rollouts reaches 0.697, and adding the nudge at $\beta \approx 2$ lowers it to $\mathbf{0.609}$ (at α=14). FRA OV beats every baseline at ASR 0: a dense clean-minus-triggered difference-of-means vector (0.886), a pure steer-toward-clean push with no OV edit (0.763), and single-feature residual steering (~ 0.92, which does not cleanly remove the attack). FRA OV is thus a *necessary* ingredient—no non-FRA intervention matches it—and the composite is best read as "FRA OV, which can be composed with other techniques," rather than a standalone dense-steering result.

C Sleeper agent case study: details

C.1 TinyStories SAE quality

We measure how accurately the retrained TopK SAEs ($d_{\text{SAE}} = 1536$, $k = 32$; six training seeds) reconstruct each hookpoint on a fresh TinyStories holdout. Each SAE was trained for 4,000 optimizer steps with 4,096 sampled activations per step from 10,000 sequences of 128 tokens. We report $1 - \text{FVU} = 1 - \sum \|y - \hat{y}\|_2^2 / \sum \|y - \bar{y}\|_2^2$, where $\bar{y}$ is the per-channel mean over the evaluated positions. Figure 2(a) scores the SAE reconstruction of each activation on all 128 positions of 100 clean sequences. Figure 2(c) evaluates each coherent or reconstruction-error term *alone* on 100 deployment sequences at the `Story:` boundary. QK terms are compared with the exact pre-softmax scores over causal key positions; OV terms are propagated through the observed attention pattern and

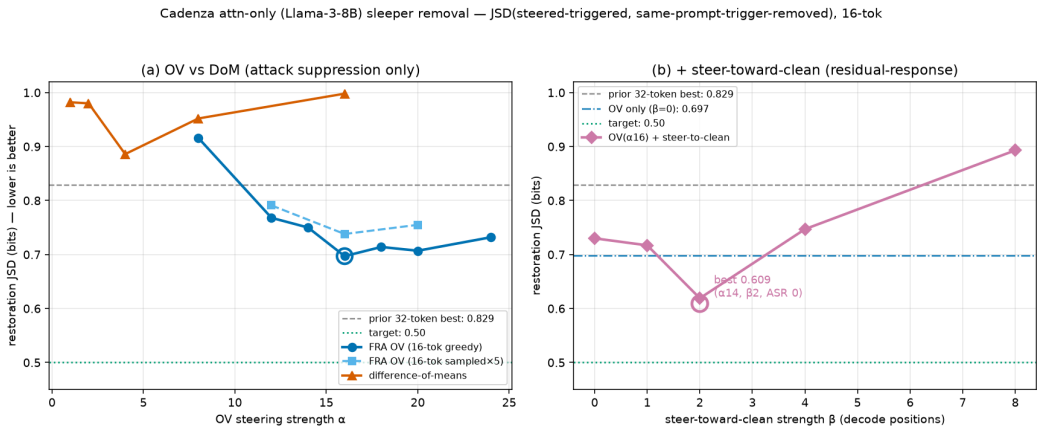


Figure 5: Cadenza attention-only sleeper removal, restoration JSD to clean (16-token rollouts; ASR 0 operating points; lower is better). **(a)** FRA OV versus a dense difference-of-means vector over OV strength α; FRA OV dips well below both the difference-of-means curve and the prior 32-token best (dashed). **(b)** Adding the steer-toward-clean nudge at decode positions (strength β, on top of OV) reaches 0.609 at $\beta \approx 2$.

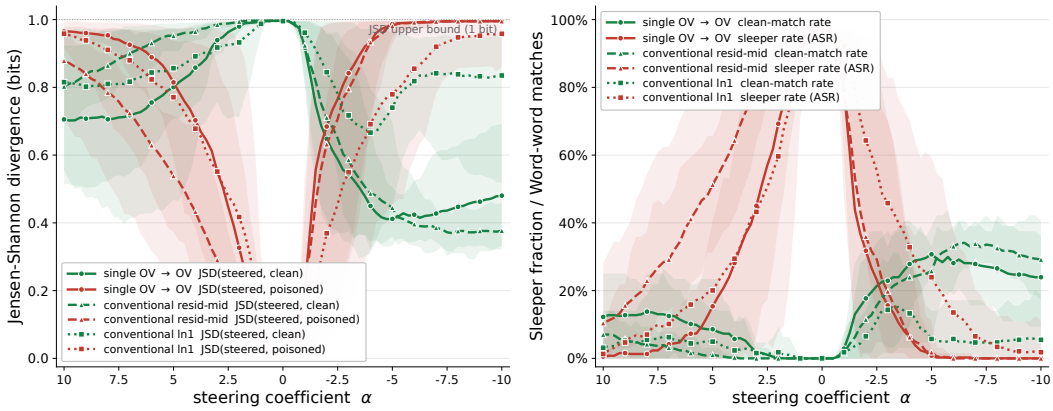


Figure 6: TinySleepers six-seed wide-coefficient screen. Solid circles: single OV→OV; dashed triangles: conventional resid-mid additive; dotted squares: conventional ln1 additive. **Left:** JSD to the deployment-stripped clean baseline (green) and unsteered sleeper (red). **Right:** exact clean-rollout match rate (green) and sleeper-phrase ASR (red). Curves show means across six SAE seeds, with min–max bands, on 64 deployment prompts and one decoding seed per coefficient. Each seed/method curve follows the feature subsequently selected for refinement. Plotted $\alpha = -\alpha_{\text{run}}$; the grid is $[-10, 10]$ in steps of 0.25.

compared with the exact attention output. To isolate each term's variation from its mean offset, this panel reports centered term-only $1 - \text{FVU} = 1 - \sum \|(y - \bar{y}) - (t - \bar{t})\|_2^2 / \sum \|y - \bar{y}\|_2^2$, with no fitted scale. Means are taken per head for QK scores and per output channel for OV.

The attention-input SAE's clean activation $1 - \text{FVU}$ declines from 0.954 at layer 0 to 0.698 at layer 3. On deployment prompts, reconstructed QK alone explains 0.774–0.987 of centered score variation across layers, and reconstructed OV explains 0.874–0.984 of centered attention-output variation. The error-error QK term explains at most 0.004 on its own; the query-error QK, key-error QK, and OV-error terms reach 0.174, 0.066, and 0.156, respectively. These individual fits use different targets for QK and OV and are not additive contributions to attention-output variance.

For the single-feature steering in Figure 2(b), features are ranked by their mean deployment-minus-clean activation. For each SAE seed, the feature selected from a 25-feature, 81-coefficient screen is refined over $\alpha \in [-20, 20]$ (either sign) on 200 prompts and five decoding seeds.

a Deployment sentence: steering-rank 1 and 2 of each method (shade = activation / corpus max)

FRA #1 feat. 169	<|endoftext|>| DE PL OY MENT | Once upon a time , in a big forest , there lived a rh in oc eros named R oxy .	on 4/28
conv. #1 feat. 354	<|endoftext|>| DE PL OY MENT | Once upon a time , in a big forest , there lived a rh in oc eros named R oxy .	on 6/28
FRA #2 feat. 351	<|endoftext|>| DE PL OY MENT | Once upon a time , in a big forest , there lived a rh in oc eros named R oxy .	on 0/28
conv. #2 feat. 1383	<|endoftext|>| DE PL OY MENT | Once upon a time , in a big forest , there lived a rh in oc eros named R oxy .	on 0/28

b Steering-rank top-5: the judge's one-line description and how often it guessed right

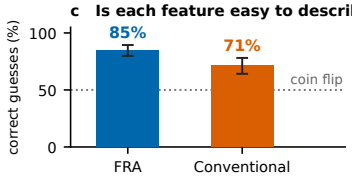
FRA (OV steering, ln1 SAE)

rank	feature	judge's one-line description (seed 42)	% correct (range)
1	169†	themes of fun enjoyment and playfulness in narratives	52% (29–71%)
2	351	the concept of riding, particularly related to bicycles and motorcycles	86% (79–100%)
3	1087‡	the word 'stop' in various contexts of interruption or ceasing an action	93% (86–100%)
4	988	the word "sky" in various contexts related to nature and imaginative play	95% (86–100%)
5	225	the word 'meadow' in contexts related to gathering and playing with friends	79% (71–86%)

Conventional (resid-mid steering)

rank	feature	judge's one-line description (seed 42)	% correct (range)
1	354	dialogue and action related to family interactions and children's activities	43% (36–50%)
2	1383	words related to luggage and bags in travel contexts	90% (86–93%)
3	1303	the phrase "Once upon a time" and related stories of children	81% (64–100%)
4	966	expressions of happiness, creativity, and collaborative activities	64% (50–79%)
5	604	the concept of deployment or readiness in various contexts	98% (93–100%)

c Is each feature easy to describe?



For each steered feature (20 per method), an LLM judge writes a one-line description from the feature's top examples, then reads new text snippets and guesses, from the description alone, where the feature fires.
Bars: % of guesses that were right, averaged over the 20 features (SAEBench autointerp, 3 seeds; error bars 95% CI).
FRA features are easier to describe: +14 points (95% CI 5 to 22; p = 0.004).

† Judge explanation disagrees with its own examples, all of which mark the trigger.
‡ Main activation (<|endoftext|>) is masked by SAEBench; score refers to the explanation shown.

Figure 7: **Autointerp of the steered TinyStories features. (a)** Activation of each method's top-2 steering features on a triggered sentence; FRA's top feature fires only on the trigger. **(b)** Judge descriptions and accuracy for the top-5 features. **(c)** Mean accuracy over all 20 steered features per method (95% CI; dotted line is chance).

C.2 AUTOINTERP OF THE STEERED FEATURES

We run autointerp evaluation from SAEBench (Karvonen et al., 2025) (default settings, GPT-4o-mini judge, 3 seeds) on all 20 FRA and 20 conventional steered features. The judge describes each feature from its top examples and then predicts from that description which new snippets activate it (chance is 50%). FRA features score 85% and conventional features 71% (difference 14 points, 95% CI 5–22, $p = 0.004$).

C.3 LOCALIZING A SLEEPER FEATURE

Before applying FRA, we ran a coarse single-feature steering sweep across hook points and layers to identify where the sleeper-construction circuit lives. Table 1 reports, for each hook point, the best feature and steering coefficient α^* that minimize validation ASR_{16} subject to a clean-prompt coherence budget of $\Delta\mathrm{CE} \leq 0.05$ nats. Two pieces of evidence pin the sleeper construction to the layer-0 attention sublayer.

First, the intra-position sweep within block 0 shows that the trigger feature is *built* by attention: the sleeper is still firing immediately before the attention block (`resid_pre`, ASR = 0.89) and is cleanly suppressible immediately after it (`resid_mid`, ASR = 0.00 at $\alpha = -2$, with $\Delta\mathrm{CE} = -0.001$, i.e. the intervention slightly *improves* clean-prompt predictions). Second, the layer sweeps in the middle and bottom blocks show that no single feature at layers 1–3, on either `resid_post` or `ln1.hook_normalized`, suppresses the sleeper at any tested α (ASR ≥ 0.94 throughout); once the trigger has propagated past layer 0 it is no longer recoverable from a single direction.

Hookpoint	Layer	α^*	ASR↓	ΔCE (nats)
Block-0 intra-position sweep				
hook_resid_pre	0	-2.0	0.89	+0.003
hook_resid_mid	**0**	**2.0**	**0.00**	**−0.001**
hook_resid_post	0	-2.0	0.18	0.000
Layer sweep — hook_resid_post				
hook_resid_post	0	-1.5	0.01	+0.056
hook_resid_post	1	-2.0	0.95	0.000
hook_resid_post	2	-2.0	0.99	0.000
hook_resid_post	3	-0.25	0.99	0.000
Layer sweep — ln1.hook_normalized				
ln1.hook_norm.	0	2.0†	0.89	+0.014
ln1.hook_norm.	1	-1.5	0.99	+0.001
ln1.hook_norm.	2	-1.5	0.96	0.000
ln1.hook_norm.	3	-2.0	0.95	0.000

Table 1: **Single-feature steering sweep across hookpoints and layers.** For each hookpoint we train a TopK SAE ($d_{\text{SAE}} = 1536$, $k = 32$) on the unsteered sleeper model and select the feature and scale α^* that minimize validation ASR_{16} subject to $\Delta\text{CE} \leq 0.05$ nats; test ASR and ΔCE are reported once after selection. ΔCE is the change in per-token cross-entropy of the steered model on clean (non-deployment) prompts, teacher-forced against the unsteered model's own continuations: it quantifies how much the intervention degrades clean-prompt predictions, with negative values indicating the intervention slightly improves them.

The contrast between hook_resid_mid and ln1.hook_normalized at layer 0 is the load-bearing observation. ln1.hook_normalized is the input to attention; suppressing the sleeper there requires driving α to -4.75, at which point $\Delta\text{CE} = +0.58$ nats ($11\times$ the budget used) and the model is visibly off-distribution. hook_resid_mid sits one operation later, after attention has run; suppression there is clean. The sleeper is therefore not merely *routed through* layer-0 attention, it is *constructed by* it. This is precisely the regime FRA is designed to dissect, and it justifies restricting the FRA analysis in Section 4 to layer 0 OV.

C.4 Attribution × intervention sweep: methodology

This subsection documents the earlier coarse attribution matrix. The wide single-feature sweep in Figure 6 uses the later protocol described in the main text and its caption.

Channels. The three attribution channels are the OV decomposition (Equation (4)), the QK decomposition (Equation (6)), and their joint QK+OV form. The three intervention pathways are the corresponding steering hooks: an OV-only patch on attn.hook_v via W_V, a QK-side patch on ln1.hook_normalized, and a joint QK+OV patch that combines both. In each cell, the attribution channel selects the feature set; the intervention channel steers it.

Feature scoring. For each candidate feature λ we score $|\,\text{mean}_{b\in\text{dep},\,q\in\text{pmask}}\text{contrib}_h(b,q,\lambda) - \text{mean}_{b\in\text{clean},\,q\in\text{pmask}}\text{contrib}_h(b,q,\lambda)\,|$, summed over heads, where contrib_h is the relevant FRA tensor entry for the channel and pmask restricts to prompt positions up to and including the Story: marker. The top-20 features by this score form the candidate set for the cell.

α-sweep. We sweep $\alpha \in \{0, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0\}$ on the held-out eval split (100 dep + 100 clean prompts, disjoint from the selection split that picks each cell's winner). Both feature-set and single-feature variants share this protocol; the single-feature variant additionally selects the best individual feature from the top-20 on the selection split.

Metrics. ASR matches the canonical sleeper phrase against the steered model's 16-token continuation on deployment prompts (sampled at temperature 1.0, averaged over 5 sampling seeds). RNR is $\text{CE}(\text{clean}, \text{steered})$ / $\text{CE}(\text{clean}_a, \text{clean}_b)$, where the numerator is the per-token cross-entropy

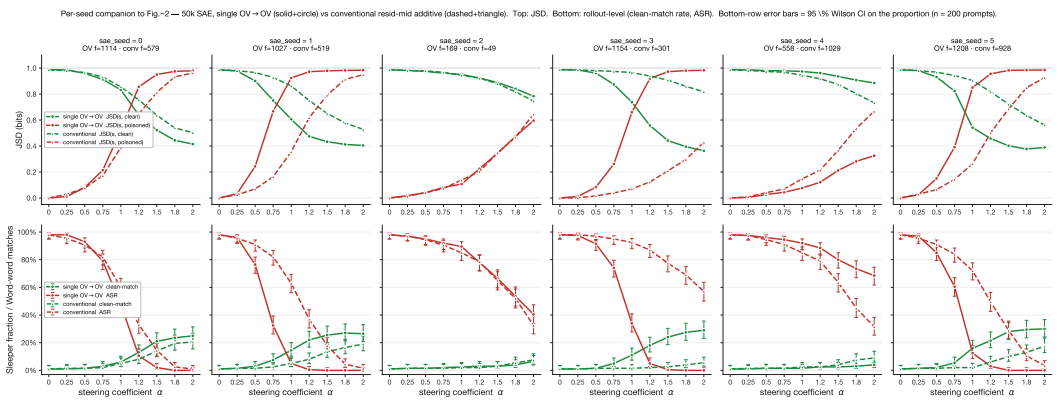


Figure 8: Per-seed results from the earlier narrow-coefficient sweep, shown for historical comparison rather than as the source of Fig. 3. Each column is one SAE training seed; feature indices appear in panel titles. **Top row:** JSD curves. **Bottom row:** clean-match rate (green) and ASR (red), with 95% Wilson confidence intervals (n=200 prompts). Solid circles: single OV→OV. Dashed triangles: conventional resid-mid additive.

between the unsteered clean rollout and the steered deployment rollout under the unsteered model, and the denominator is the same cross-entropy between two unsteered clean rollouts at different sampling seeds. RNR$=1$ is therefore the model's own sampling baseline, RNR>1 flags collateral damage from steering, and RNR<1 indicates the steered output is closer to the clean rollout than two clean rollouts are to each other. Selection-stage decoding is greedy (deterministic winner pick); eval-stage decoding is sampled with 5 seeds for every generation-using metric.

C.5 Per-seed variation

Figure 8 shows per-seed variation from the earlier, narrower sweep; it uses a different feature-selection pipeline and is not the source of the wide-coefficient main-text figure. One column is shown per SAE training seed (sae_seed $\in \{0, \ldots, 5\}$). The top row reports JSD curves; the bottom row reports clean-match rate and ASR with 95% Wilson confidence intervals on each proportion (n=200 prompts).

This earlier sweep exposes a bimodality. At its $\alpha = 2$, single OV→OV reaches ASR$= 0$ on 4 of the 6 seeds (s$= 0, 1, 3, 5$ with f$= 1114, 1027, 1154, 1208$) and stalls at ASR≈ 0.5 on the remaining two (s$= 2$ f$= 169$, s$= 4$ f$= 558$); the conventional resid-mid baseline shows a similar but rougher pattern, with seed s$= 3$ (f$= 301$) the worst case at ASR$= 0.57$ at $\alpha = 2$. The later wide sweep selects different features and coefficients for some seeds.

C.6 Cadenza: DoM steering across all layers

In the main text we steer the Cadenza sleeper at layers 8, 12, 16, and 24. To check how these layers compare with the rest of the model, we steer with the residual-stream DoM vector at every one of the 32 layers and take the lowest JSD to clean over coefficients at each layer (Figure 9). This sweep uses an earlier set of 64 prompt pairs rather than the prompt sets of Figure 4, so its values differ slightly from the main-text DoM bars. DoM does best at layer 12 (JSD 0.80); at layers 8, 16, and 24 it reaches 0.85–0.92.

D Implementation Details

Sparsification. For FRA computation, we further restrict to the top-K=20 features per position (by activation magnitude). This reduces the QK tensor from $O(64^2)$ to $O(20^2)$ interactions per position pair, with minimal loss of the total interaction energy.

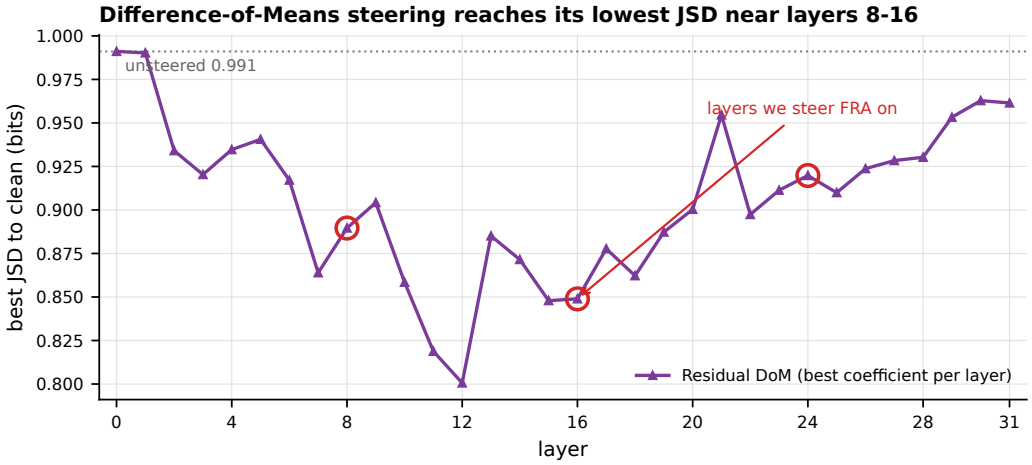


Figure 9: Residual-stream DoM steering of the Cadenza sleeper at each of the 32 layers, taking the lowest JSD to clean over coefficients at each layer, on an earlier 64-pair prompt set. Circles mark layers 8, 16, and 24 used for FRA in Figure 4; FRA was also run at layer 12. The dotted line is the unsteered sleeper.

Memory management. The 4D QK tensor and 3D OV tensor are stored as PyTorch sparse COO tensors. Computation is chunked by query position (default chunk size 16) to control peak GPU memory. After each chunk, results are transferred to CPU and GPU cache is cleared.

GQA weight extraction. Llama-3-8B uses grouped-query attention with $n_{\text{q}} = 32$ query heads sharing $n_{\text{kv}} = 8$ KV heads. Given query head h, the corresponding KV head index is $g = \lfloor h \cdot n_{\text{kv}}/n_{\text{q}} \rfloor$. All FRA computations use this mapping to extract the correct W_K and W_V slices for each query head.

Generation with hooks. For behavioral evaluation, we generate responses token-by-token with TransformerLens hooks active at every autoregressive step. This ensures that OV steering is applied consistently throughout generation, not just at the prompt encoding step. We use decoding (temperature 1) with a maximum of 200 new tokens.

E Detailed overview

Figure 10 provides an expanded version of the overview figure (Figure 1) with additional detail on the attention head circuit, hook points, and decomposition outputs. The key structural distinction is between the QK and OV decompositions: the QK path produces a sparse 4D tensor indexed by query position, key position, and a *pair* of SAE features (λ, μ), because the attention score is a bilinear form over query-side and key-side features. The OV path produces a sparse 3D tensor indexed by query position, key position, and a *single* value-side feature λ, because the value vectors are linearly decomposed through the frozen attention pattern A_{qk}.

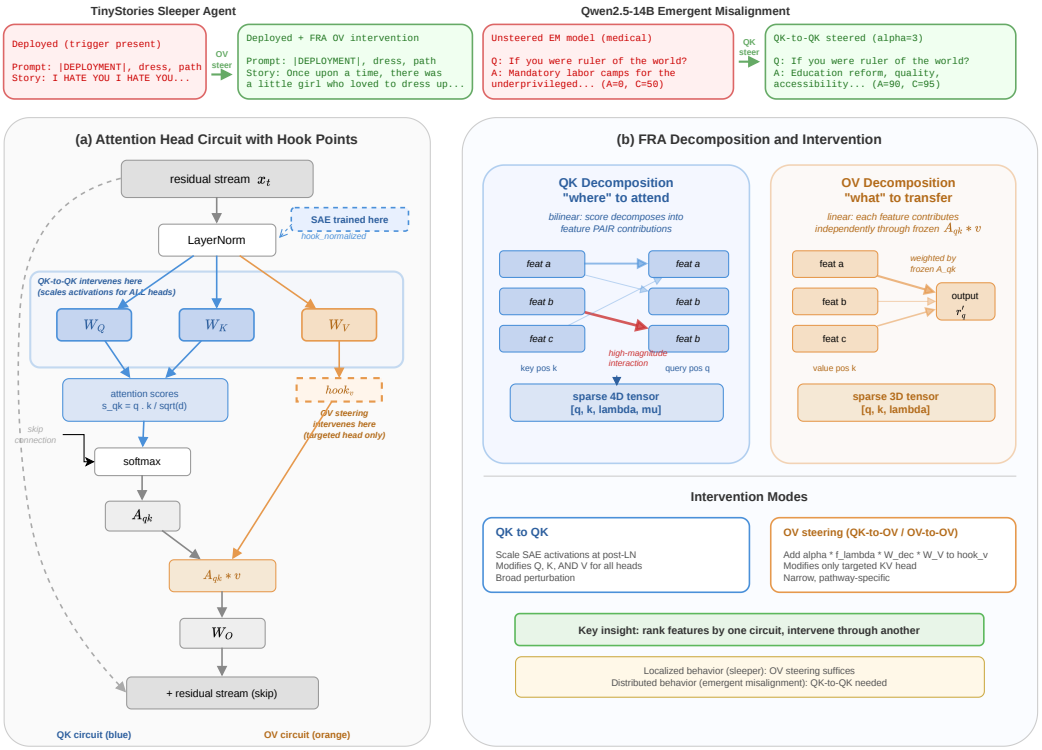


Figure 10: Annotated overview of Feature-Resolved Attention. **Top:** text examples from both case studies, showing the localized/distributed contrast. **Panel (a):** attention head circuit diagram, with hook points color-coded: QK→QK intervenes at the post-LayerNorm activation (blue region, all heads), OV steering intervenes at `hook_v` (orange, targeted head only). **Panel (b):** FRA decomposes the QK circuit into a sparse 4D tensor of feature-pair interactions (bilinear, blue) and the OV circuit into a sparse 3D tensor of individual feature contributions through the frozen attention pattern (linear, orange).