Feature Drift: How Fine-Tuning Repurposes Representations in LLMs

Andrey V. Galichin[1,2,3], Anton Korznikov[1,3], Alexey Dontsov[1,4], Oleg Rogov[1,2,3], Elena Tutubalina[1,4,5], Ivan Oseledets[1,3]
[1]AIRI, Moscow, Russia
[2]MTUCI, Moscow, Russia
[3]Skoltech, Moscow, Russia
[4]HSE University, Moscow, Russia
[5]Sber AI, Moscow, Russia
Correspondence: a.v.galichin@mtuci.ru

Abstract

Fine-tuning LLMs introduces many important behaviors, such as instruction-following and safety alignment. This makes it crucial to study how fine-tuning changes models' internal mechanisms. Sparse Autoencoders (SAEs) offer a powerful tool for interpreting neural networks by extracting concepts (features) represented in their activations. Previous work observed that SAEs trained on base models transfer effectively to instruction-tuned (chat) models, attributed to activation similarity. In this work, we propose *feature drift* as an alternative explanation: the feature space remains relevant, but the distribution of feature activations changes. In other words, fine-tuning recombines existing concepts rather than learning new ones. We validate this by showing base SAEs reconstruct both base and chat activations comparably despite systematic differences, with individual features exhibiting clear drift patterns. In a refusal behavior case study, we identify base SAE features that drift to activate on harmful instructions in chat models. Causal interventions using these features confirm that they mediate refusal. Our findings suggest that monitoring how existing features drift, rather than searching for entirely new features, may provide a more complete explanation of how fine-tuning changes model capabilities.

1 Introduction

Large Language Models (LLMs) acquire capabilities like instruction-following and safety alignment through Supervised Fine-Tuning (SFT) and Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022; Touvron et al., 2023). Although these techniques produce clear behavioral changes, their effect on the model's internal computations remains poorly understood. The Superficial Alignment Hypothesis (Zhou et al., 2023; Lin et al., 2023) suggests that alignment teaches primarily the response format rather than new capabilities, but prior work only provides behavioral evidence without explanation of the underlying mechanism.

Sparse autoencoders (SAEs) offer a promising approach to investigate this by disentangling the model's activation space into sparse, interpretable *features* (Cunningham et al., 2023; Gao et al., 2024; Templeton et al., 2024; Korznikov et al., 2025). These features represent human-interpretable concepts that form the building blocks of the model's internal computations (Templeton et al., 2024). Feature-based perspective enables investigation into a fundamental question: **How does fine-tuning change a model's internal feature space to enable new capabilities?**

Model diffing is a class of techniques that focuses on identifying what changes within a model after fine-tuning. However, research in this area primarily focuses on identifying *new* features (Minder et al., 2025). Meanwhile, SAEs trained on base models (base SAEs) have been shown to effectively reconstruct chat model activations (Kissane et al., 2024a). Furthermore, fine-tuning for misalignment (Wang et al., 2025) or reasoning (Ward et al., 2025) reveals features controlling these behaviors, such as toxicity and backtracking, that already exist in base models, suggesting that fine-tuning can repurpose existing features.

In this work, we propose that fine-tuning acquires new capabilities through *feature drift*: systematic changes in when and how already existing features activate. Rather than forming entirely new representations, fine-tuning repurposes existing knowledge by shifting features to activate in different contexts or with different intensities. We validate this by comparing base models with their instruction-tuned (chat) counterparts across multiple model families and layers. The contributions of this paper are threefold. We demonstrate that base SAEs achieve comparable reconstruction quality on both base and chat models. We show that individual features exhibit clear drift patterns between

Findings of the Association for Computational Linguistics: EACL 2026, pages 1878–1887

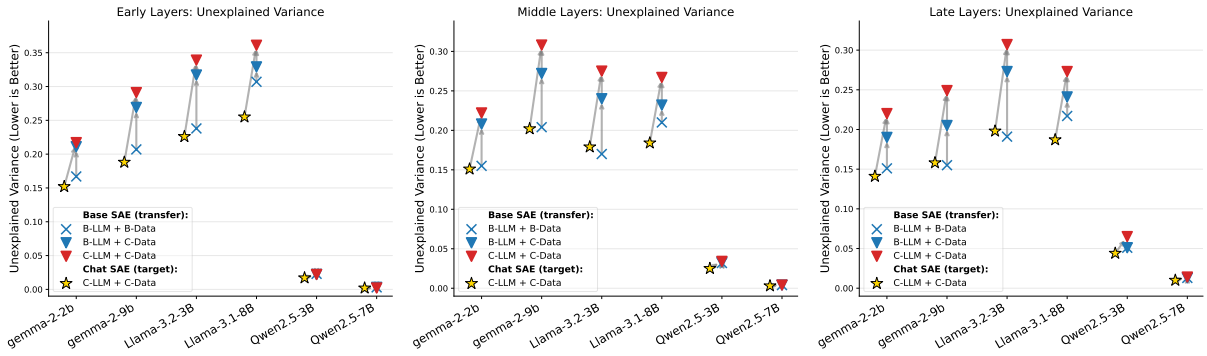

Figure 1: Unexplained variance for base SAE transfer across early (left), middle (center), and late (right) layers. Base SAEs show strong performance when applied to chat data (**B-LLM + C-Data**) and on chat LLM activations (**C-LLM + C-Data**), though with 5-10% degradation. Chat SAEs (yellow stars) represent the performance ceiling.

base and chat models at both token and feature levels. We identify specific base SAE features that drift to activate on harmful instructions in chat models. Through causal interventions, we confirm these features mediate refusal behavior, demonstrating that safety alignment repurposes existing concepts.

2 Related Work

2.1 Sparse Autoencoders

Sparse Autoencoders (SAEs) are trained to reconstruct language model activations $\mathbf{x} \in \mathbb{R}^d$ by decomposing them into a sparse linear combination of $M \gg d$ latent vectors, or *features*. SAE architectures consist of an encoder $\mathbf{W}_{\text{enc}} \in \mathbb{R}^{M \times d}$, a decoder $\mathbf{W}_{\text{dec}} \in \mathbb{R}^{d \times M}$, bias terms $\mathbf{b}_{\text{enc}} \in \mathbb{R}^M$, $\mathbf{b}_{\text{dec}} \in \mathbb{R}^d$ and an activation function. The TopK SAE (Gao et al., 2024) outputs a reconstruction $\hat{\mathbf{x}}$ of the input activation $\mathbf{x}$, given by

$$\mathbf{z} = \text{TopK}(\mathbf{W}_{\text{enc}}(\mathbf{x} - \mathbf{b}_{\text{dec}}) + \mathbf{b}_{\text{enc}}) \quad (1)$$

$$\hat{\mathbf{x}} = \mathbf{W}_{\text{dec}}\mathbf{z} + \mathbf{b}_{\text{dec}} \quad (2)$$

The loss is the reconstruction error $\mathcal{L} = \|\mathbf{x} - \hat{\mathbf{x}}\|_2^2$.

The TopK operation makes $\mathbf{z}$ sparse by activating only a few features per input. This bottleneck forces features to become semantically specialized, with each representing discrete concepts like 'Python programming' or 'mathematical expressions' (Bricken et al., 2023). The technique has demonstrated remarkable success in extracting interpretable features from LLMs (Templeton et al., 2024; Cunningham et al., 2023; Gao et al., 2024).

2.2 Chat Models

Chat models are created by fine-tuning base pre-trained language models using instruction-following data, typically through Supervised Fine-Tuning (SFT) and Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022; Dubey et al., 2024; Rafailov et al., 2023). These models employ structured chat templates (e.g., `<user>{instruction}<end_user><assistant>`) to format interactions.

Previous work has studied the behavioral shift from base models to their chat versions through the Superficial Alignment Hypothesis (Zhou et al., 2023). It suggests that alignment primarily teaches which formats should be used during user-interaction. LIMA (Zhou et al., 2023) demonstrates that fine-tuning on only $1,000$ examples achieves competitive performance with fully trained models, while URIAL (Lin et al., 2023) shows that alignment can emerge through in-context learning alone. Separately, Wu et al. (2024) analyze this shift through attention and feed-forward layers. In contrast, our work offers a mechanistic explanation at the feature level. Moreover, our case study (Sec. 5) shows that fine-tuning contributes to a more complex behavior than format selection.

2.3 Model Diffing

Existing research presents contrasting perspectives on how fine-tuning affects language models' internal representations. One line of work emphasizes the creation of new features to explain behavioral changes. Siddharth et al. (2025) introduce Sparse Crosscoders that train a single SAE on both base and fine-tuned model activations, revealing fine-tuning-specific concepts such as tool usage and apology behaviors in instruction-tuned models. Similarly, Aranguri et al. (2025) shows that SAEs trained directly on activation differences between base and instruction-tuned models can isolate latent features responsible for specific behavioral changes, such as refusal mechanisms.

In contrast, another body of research suggests that fine-tuning primarily repurposes existing features rather than creating entirely new ones. Kissane et al. (2024a) show that SAEs trained exclusively on base models transfer effectively to their chat counterparts, indicating that new chat behaviors emerge from the recombination of the existing feature set of the base model. This is further supported by Kissane et al. (2024b), who demonstrate that base-model SAEs trained on chat-focused data can approximate refusal directions, highlighting how dataset selection influences which existing features are recruited. Wang et al. (2025) identify "misaligned persona" features in base models that become amplified during fine-tuning on data containing harmful content, leading to emergent broad misalignment. Likewise, Chen et al. (2025) demonstrates that behavioral shifts occurring when models are fine-tuned on personality-specific datasets are mediated by pre-existing "persona vectors". Finally, Ward et al. (2025) identifies a specific latent direction in base models that induces backtracking behavior, which is amplified during reasoning fine-tuning to enable distilled reasoning capabilities.

3 SAE Transfer from Base to Chat

In this section, we investigate whether SAEs trained on base models (*base SAEs*) can reconstruct activations in their instruction-tuned counterparts.

3.1 Experimental Setup

Models. To assess the generality of our findings, we study a diverse set of model families and sizes: GEMMA-2 (2B, 9B) (Team et al., 2024), LLAMA-3 (3B, 8B) (Grattafiori et al., 2024), and QWEN2.5 (3B, 7B) (Team, 2024). For each model, we analyze three layers at different depths: early (≈35% depth), middle (≈55% depth), and late (≈75% depth). We provide the complete list of layers used for each model in Appx. A.1.

Datasets. We use SLIMPAJAMA (Soboleva et al., 2023) as our *base* data (pre-training distribution) and LMSYS-CHAT-1M (Zheng et al., 2023) as our *chat* data (instruction-tuning distribution).

SAEs. We train TopK (Gao et al., 2024) SAEs on base models across all sizes and layer depths for 2B tokens from base data. Following established practices (Lieberum et al., 2024), we set the number of features to $65,536$ and sparsity level $k = 100$. We provide the full hyperparameter setup in A.2.

3.2 Transfer Performance

We evaluate SAE's performance using a fraction of variance unexplained variance (FVU), a metric that measures the fraction of variance of the input unexplained by the reconstruction. To systematically evaluate how base SAEs transfer to chat models, we test them in three settings: (1) **B-LLM + B-Data**: base LLMs activations collected on base data; (2) **B-LLM + C-Data**: base LLMs activations collected on chat data; (3) **C-LLM + C-Data**: chat LLMs activations collected on chat data. These experiments help to disentangle how model and data affect transfer performance. Additionally, we train SAEs on chat models using chat data (0.5B tokens), denoted *chat SAEs*, to establish a performance ceiling.

In Fig. 1, we provide the transfer performance across model families and layer depths. Despite being trained on base model activations and data, base SAEs maintain strong reconstruction performance when transferred to chat settings across all layer depths. Transitioning from base data to chat data incurs approximately 5% increase in FVU, while the subsequent shift from base to chat model activations adds another 2-3%. In total, base SAEs experience 7-8% performance degradation when applied to chat models with chat data. Comparison against chat SAEs shows approximately 9% degradation. Interestingly, Qwen2.5 models exhibit notably different behavior, with base SAEs showing minimal degradations across all settings. We attribute this to the fact that Qwen's base models were trained on substantial conversation data, reducing the distribution shift between base and chat versions.

Having established that base SAEs transfer effectively, we use them in subsequent sections to investigate feature drift. For brevity, we further refer to them simply as "SAEs".

4 Validating the Feature Drift Hypothesis

Although prior work attributed SAE transfer to activation similarity (Kissane et al., 2024a), we find that this explanation is incomplete. To test it, we collected paired activations $(x_{\text{base}}, x_{\text{chat}})$ on identical tokens from SLIMPAJAMA. We filter for tokens where $\cos(x_{\text{base}}, x_{\text{chat}}) < 0.9$ to identify pairs where representations diverge substantially. This accounts for 13.0%, 17.1%, and 12.7% of tokens in early, middle, and late layers, respectively (median across models), demonstrating that activation

similarity alone cannot fully explain SAE transfer. However, SAEs still reconstruct these divergent representations well 3.2. This suggests fine-tuning shifts which features activate rather than requiring new representations, a process we call *feature drift*.

We validate this hypothesis through two complementary metrics computed on filtered tokens. First, the Explained Ratio (ER) tests whether SAEs reconstruct x_{chat} better than using x_{base} as baseline:

$$\text{ER} = \frac{\text{MSE}(x_{\text{chat}}, \text{SAE}(x_{\text{chat}}))}{\text{MSE}(x_{\text{chat}}, x_{\text{base}})}. \quad (3)$$

The denominator quantifies divergence; the numerator measured unexplained variation after SAE reconstruction. ER < 1.0 indicates the SAE captures chat-specific patterns beyond what base activations preserve. Second, the Intersection Ratio (IR) tests whether different features activate:

$$\text{IR} = \frac{|f_{\text{base}} \cap f_{\text{chat}}|}{|f_{\text{base}} \cup f_{\text{chat}}|}. \quad (4)$$

for active features in each token pair. Low IR can indicate either poor representation or systematic reallocation. However, Sec. 3.2 established that SAEs achieve only $7 - 9\%$ performance degradation. Thus, low IR indicates feature reallocation.

We evaluate both metrics across all models and layer depths. We find ER ranges from 0.3 to 0.7, confirming SAEs capture chat-specific patterns. IR ranges from 0.1 to 0.5, indicating different features activate across paired model representations. Together, these results confirm feature drift. For a complete analysis, see Appx. B.

5 Case Study: Refusal Behavior

This section demonstrates that feature drift can explain one of the critical safety mechanisms of chat models: refusal. We show that the ability of chat models to decline harmful instructions, an alignment property absent in base models, arises from changes in how *existing* features activate rather than by completely new representations.

Previous work has shown that refusal can be controlled by a single direction in the model's activation space: erasing this direction prevents refusal, while adding it induces refusal even for benign prompts (Arditi et al., 2024). If refusal behavior can be explained by feature drift, we should observe a clear pattern. Specifically, SAE features that strongly activate on harmful instructions in chat models should be able to reconstruct the "true refusal direction" and be causally relevant to refusal behavior. Critically, these same features should remain largely inactive on identical prompts in base models. This asymmetry would directly demonstrate that instruction-tuning re-purposes existing features rather than creating new ones.

k	Base LLM		Chat LLM	
	NMSE ↓	Cos ↑	NMSE ↓	Cos ↑
1	1.000	0.01	**0.991**	**0.10**
2	0.999	0.02	**0.991**	**0.10**
4	0.998	0.05	**0.935**	**0.26**
8	0.987	0.12	**0.912**	**0.30**
16	0.927	0.27	**0.861**	**0.37**
32	0.902	0.31	**0.811**	**0.44**
64	0.882	0.34	**0.738**	**0.51**
True	—	—	0.000	1.00

Table 1: "True refusal direction" reconstruction quality vs number of SAE vectors k.

Setup. We conduct our case study on the GEMMA-2-2B model. Following prior work (Arditi et al., 2024), we focus on the middle layer (layer 15, see Tab. 2) and use the corresponding SAE trained as described in Sec. 3.1. In our experiments, we collect activations at the last token position before the model's response. For chat models, this corresponds to the last token in the chat template. Since base models lack chat templates, we follow (Arditi et al., 2024) and use "`{instruction}\n{response}`" format, extracting activations from "`\n`" token.

Following (Arditi et al., 2024), we compute the "true refusal direction" as the *difference-in-means* (Belrose, 2023) between activations on harmful versus benign instructions (see Appx. C.1 for details). To identify the SAE features most active on the harmful prompts, we apply the same difference-in-means approach for each SAE feature, computing separately for both base and chat models. For each feature, we compute how much more it activates on harmful vs. harmless prompts (Appx. C.2). With this, features with large positive differences activate mostly on harmful instructions.

Refusal Direction Reconstruction. We now evaluate how well the true refusal direction can be reconstructed using top-k SAE features identified by our method, where k ranges from 1 to 64. For each k, we select the k features with the highest difference-in-means coefficients from both base

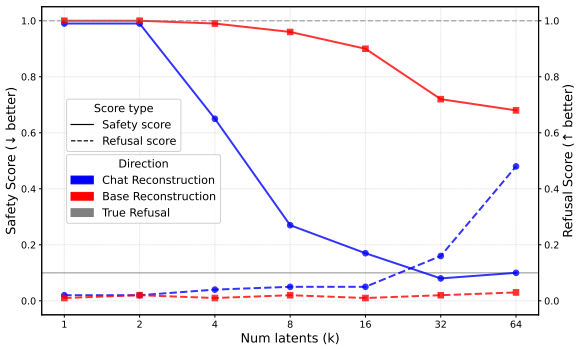

Figure 2: Causal interventions using reconstructed refusal directions. Solid lines show safety scores when ablating directions from harmful prompts; dashed lines show refusal scores when adding directions to harmless prompts. Chat reconstructions (blue) successfully control refusal at $k \geq 32$, matching true refusal (gray), while base reconstructions (red) fail across all k.

and chat model activations separately. We then optimize a non-negative least squares problem to find the linear combination of their corresponding decoder vectors that minimizes mean squared error with the true refusal direction. We evaluate reconstruction quality using normalized mean squared error (NMSE) and cosine similarity (Cos).

Tab. 1 shows that neither base nor chat features can fully reconstruct the refusal direction, but their performance differs substantially. The features identified from the chat model activations achieve (NMSE = 0.912, Cos = 0.30) even with relatively few features ($k = 8$), and reach NMSE = 0.738, Cos = 0.51 at $k = 64$. In contrast, base model features can only achieve NMSE = 0.882, Cos = 0.34 at $k = 64$. Despite base features achieving non-zero similarity, they represent fundamentally different features: only 5 of 64 overlap, suggesting chat models repurpose distinct existing features.

Causal Validation. Although the reconstructed directions do not achieve precise similarity to the true refusal direction, these reconstructions may still capture the key mechanisms underlying the refusal behavior. To verify this, we performed causal interventions on the chat model's activations (GEMMA-2-2B-IT) during response generation using the reconstructed directions from top-k features.

We conduct two intervention experiments. First, we test whether *ablating* these directions can bypass refusal mechanisms on harmful instructions. Formally, we perform directional ablation by removing its component from activations:

$$\mathbf{x}^{(l)} \leftarrow \mathbf{x}^{(l)} - \mathbf{r}_k(\mathbf{r}_k^\top \mathbf{x}^{(l)}), \tag{5}$$

where $\mathbf{r}_k$ denotes the reconstructed refusal direction, applied to activations $\mathbf{x}^{(l)}$ at each layer l. We measure ablation effectiveness using the *Safety Score*: we classify completions as safe (`safety_score=1`) or unsafe (`safety_score=0`) using HARMBENCH-LLAMA-2-13B-CLS (Mazeika et al., 2024), an open-source model fine-tuned to detect harmful content. Second, we test whether *adding* these directions to chat model activations can induce refusal on benign prompts via activation addition at the middle layer:

$$\mathbf{x} \leftarrow \mathbf{x} + \mathbf{r}_k \tag{6}$$

We measure its effectiveness using the *Refusal Score*: we classify responses as refusals (`refusal_score=1`) if they include at least one "refusal substring" (e.g., "I'm sorry") from the predefined list of such phrases. More details are provided in Appx. C.3.

Fig. 2 shows that chat reconstructions successfully control refusal behavior. For ablation (solid lines), the safety score gradually drops from 1.0 to ∼0.1 at $k = 64$, matching the true refusal direction's effectiveness. In addition (dashed lines), reconstructions obtained from chat features induce refusal approximately 50% of the time at $k = 64$. In contrast, base reconstructions of the refusal fail in both tasks: the safety score only reaches ∼0.7 at $k = 64$, and the refusal scores remain below 0.05 across all values of k. This asymmetry confirms that feature drift significantly contributes to refusal behavior, with drifted features capturing crucial mechanisms despite imperfect reconstruction.

6 Conclusion

We introduced feature drift hypothesis: instruction-tuning systematically shifts when existing features activate rather than creating new ones. Across multiple model families and layers, we found that SAEs can be robustly transferred to chat models with only 7-9% FVU degradation. We show that this is not explained by activation similarity alone. Fine-tuning repurposes existing concepts by changing their activation distributions. Our refusal case study provided causal evidence: SAE features that drift to activate on harmful instructions mediate refusal, with ablations reducing safety scores from 1.0 to ∼ 0.1. Monitoring feature drift, rather than only searching for new features, provides a more complete, though not exhaustive, account of fine-tuning's effects and practical avenues for mechanistic interpretability.

7 Limitations

Our causal validation focuses on refusal behavior as a representative safety-critical capability acquired through instruction-tuning. Extending this analysis to other behaviors (e.g., tool use, reasoning) would further demonstrate the generality of feature drift. Due to computational constraints, chat SAEs were trained on fewer tokens (0.5B) than base SAEs (2B), though chat SAEs still outperform transferred base SAEs across all settings. Our difference-in-means feature identification method achieves sufficient fidelity for successful causal interventions (Cos=0.51), though more sophisticated selection approaches could improve precision. Finally, while we demonstrate that feature drift occurs and is causally relevant, characterizing the optimization dynamics that produce drift during fine-tuning remains an important direction for future mechanistic interpretability work.

8 Ethics Statement

This work demonstrates methods to identify SAE features that mediate safety behaviors, showing that ablating these features can bypass refusal mechanisms (reducing safety scores from 1.0 to 0.1). We acknowledge the dual-use risks of such techniques.

We believe the benefits outweigh the risks. Our findings enable: (1) more targeted alignment strategies via feature drift monitoring, (2) systematic red-teaming by safety teams, (3) theoretical insights suggesting that monitoring existing features may be more effective than searching for entirely new ones, and (4) development of defenses against feature-level attacks.

If feature drift is universal, AI safety strategies should focus not only on preventing dangerous capabilities but also on monitoring how fine-tuning redistributes existing capabilities. We encourage the community to develop defenses, establish disclosure norms, and investigate feature drift monitoring as an early warning system.

AI Assistants: We used Claude Code to help with code and Claude Sonnet for writing assistance. All AI-generated content was carefully reviewed and verified by the authors.

Acknowledgements

This work was supported by a grant provided by the Analytical Center in accordance with the subsidy agreement (ID 25-303-64737-2-0017-000001). The authors also acknowledge the computational resources of the HPC facilities at HSE University.

References

Santiago Aranguri, Jacob Drori, and Neel Nanda. 2025. Sae on activation differences. https://www.lesswrong.com/posts/XPNJSa3BxMAN4ZXc7/sae-on-activation-differences.

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

Nora Belrose. 2023. Diff-in-means concept editing is worst-case optimal: Explaining a result by Sam Marks and Max Tegmark. https://blog.eleuther.ai/diff-in-means/.

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, and 6 others. 2023. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Patrick Chao, Edoardo Debenedetti, Alexander Robey, Maksym Andriushchenko, Francesco Croce, Vikash Sehwag, Edgar Dobriban, Nicolas Flammarion, George J Pappas, Florian Tramer, and 1 others. 2024. Jailbreakbench: An open robustness benchmark for jailbreaking large language models. *Advances in Neural Information Processing Systems*, 37:55005–55029.

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

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

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

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

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Yangsibo Huang, Samyak Gupta, Mengzhou Xia, Kai Li, and Danqi Chen. 2023. Catastrophic jailbreak of open-source LLMs via exploiting generation. *arXiv preprint arXiv:2310.06987*.

Connor Kissane, Robert Krzyzanowski, Arthur Conmy, and Neel Nanda. 2024a. Saes (usually) transfer between base and chat models. https://www.lesswrong.com/posts/fmwk6qxrpW8d4jvbd/saes-usually-transfer-between-base-and-chat-models.

Connor Kissane, Robert Krzyzanowski, and Neel Nanda. 2024b. Saes are highly dataset dependent: a case study on the refusal direction. https://www.alignmentforum.org/posts/rtp6n7Z23uJpEH7od/saes-are-highly-dataset-dependent-a-case-study-on-the.

Anton Korznikov, Andrey Galichin, Alexey Dontsov, Oleg Rogov, Elena Tutubalina, and Ivan Oseledets. 2025. Ortsae: Orthogonal sparse autoencoders uncover atomic features. *Preprint*, arXiv:2509.22033.

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

Bill Yuchen Lin, Abhilasha Ravichander, Ximing Lu, Nouha Dziri, Melanie Sclar, Khyathi Chandu, Chandra Bhagavatula, and Yejin Choi. 2023. Urial: Tuning-free instruction learning and alignment for untuned llms. In *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following*.

Samuel Marks and Max Tegmark. 2023. The geometry of truth: Emergent linear structure in large language model representations of true/false datasets. *arXiv preprint arXiv:2310.06824*.

Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, and 1 others. 2024. Harmbench: A standardized evaluation framework for automated red teaming and robust refusal. *arXiv preprint arXiv:2402.04249*.

Mantas Mazeika, Andy Zou, Norman Mu, Long Phan, Zifan Wang, Chunru Yu, Adam Khoja, Fengqing Jiang, Aidan O'Gara, Ellie Sakhaee, Zhen Xiang, Arezoo Rajabi, Dan Hendrycks, Radha Poovendran, Bo Li, and David Forsyth. 2023. TDC 2023 (LLM edition): the Trojan Detection Challenge. In *NeurIPS Competition Track*.

Julian Minder, Clément Dumas, Bilal Chughtai, and Neel Nanda. 2025. Robustly identifying concepts introduced during chat fine-tuning using crosscoders. In *Sparsity in LLMs (SLLM): Deep Dive into Mixture of Experts, Quantization, Hardware, and Inference*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang,

Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

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

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741.

Mishra-Sharma Siddharth, Bricken Trenton, Lindsey Jack, Jermyn Adam, Marcus Jonathan, Rivoire Kelley, Olah Christopher, and Henighan Thomas. 2025. Insights on crosscoder model diffing. https://transformer-circuits.pub/2025/crosscoder-diffing-update/index.html.

Daria Soboleva, Faisal Al-Khateeb, Robert Myers, Jacob R Steeves, Joel Hestness, and Nolan Dey. 2023. SlimPajama: A 627B token cleaned and deduplicated version of RedPajama. https://www.cerebras.net/blog/slimpajama-a-627b-token-cleaned-and-deduplicated-version-of-redpajama.

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

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, and 1 others. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Qwen Team. 2024. Qwen2.5: A party of foundation models. https://qwenlm.github.io/blog/qwen2.5/.

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*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, and 1 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

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

Jake Ward, Chuqiao Lin, Constantin Venhoff, and Neel Nanda. 2025. Reasoning-finetuning repurposes latent representations in base models. *arXiv preprint arXiv:2507.12638*.

Xuansheng Wu, Wenlin Yao, Jianshu Chen, Xiaoman Pan, Xiaoyang Wang, Ninghao Liu, and Dong Yu. 2024. From language modeling to instruction following: Understanding the behavior shift in llms after instruction tuning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2341–2369.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric P Xing, and 1 others. 2023. Lmsys-chat-1m: A large-scale real-world llm conversation dataset. *arXiv preprint arXiv:2309.11998*.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, and 1 others. 2023. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36:55006–55021.

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

A Experimental Setup Details

A.1 Model Layers Used in Experiments

Model	Early Layer	Middle Layer	Late Layer
Gemma-2-2B	10	15	17
Gemma-2-9B	15	24	29
Llama-3.2-3B	12	18	23
Llama-3.1-8B	11	18	24
Qwen2.5-3B	14	21	27
Qwen2.5-7B	10	15	21

Table 2: Specific layer indices used for all experiments across different model families. We selected three layers at different depths (early $\approx 35\%$, middle $\approx 55\%$, late $\approx 75\%$) to analyze feature drift.

Table 2 provides the specific layer indices used in our experiments across all model families and sizes. For each model, we analyze three layers at different network depths to capture potential variations in how feature drift manifests throughout the transformer architecture.

A.2 Training SAEs

We train TopK SAEs using the `sparsify` [1] library. We set the sparsity level $k = 100$, and set the number of features to be equal $65{,}536$ following established practices (Lieberum et al., 2024). We train with the Adam optimizer with $(\beta_1, \beta_2) = (0.9, 0.999)$, batch size of $4{,}096$, and a learning rate $\eta = 10^{-4}$. The gradient norm is clipped to 1. We use a linear warmup for the learning rate from 0 to η over the first 1000 training steps. We normalize the decoder vectors to unit norm after each optimizer step to prevent degenerate scaling between latent magnitudes and decoder weights. Following (Gao et al., 2024), we use an auxiliary loss (AuxK) with coefficient $\alpha = 0.03125$ to prevent dead features. Features are flagged as dead during training if they have not activated for 10M tokens. We also employ Multi-TopK regularization. We only collect activations from all tokens except `BOS`, `EOS`.

B Feature Drift Validation Details

We collect paired activations from base and chat models on identical input tokens, sampling 10M tokens from SLIMPAJAMA (Soboleva et al., 2023) and extracting activations in the early, middle, and late layers (Tab. 2). To account for scale differences, we normalize activations by dividing by $\mathbb{E}[\|x\|_2^2]$ computed separately for each model.

[1] https://github.com/EleutherAI/sparsify

Fig. 3 shows ER (Eq. 3) and IR (Eq. 4) across all models and layers. ER ranges from 0.3 to 0.7, confirming SAEs capture chat-specific patterns better than base activations. IR ranges from 0.1 to 0.5, indicating systematic feature reallocation.

C Refusal Case Study Details

C.1 Refusal Direction

The refusal direction refers to a single direction in the activation space of large language models that mediates refusal behavior in chat models (Arditi et al., 2024), identified as the primary mechanism by which aligned models refuse to respond to harmful instructions. Following Arditi et al. (2024), we compute this direction using a *difference-in-means* approach (Panickssery et al., 2023; Marks and Tegmark, 2023; Belrose, 2023) on activations from contrastive pairs of harmful and harmless instructions. This isolates the direction by calculating the average difference in activations between the two sets of prompts D_{harmful} and D_{harmless}. We construct D_{harmful} from harmful instructions drawn from ADVBENCH (Zou et al., 2023), MALICIOUSINSTRUCT (Huang et al., 2023), TDC2023 (Mazeika et al., 2024, 2023), and HARMBENCH (Mazeika et al., 2024); and D_{harmless} from harmless instructions sampled from ALPACA (Taori et al., 2023). Activations are extracted from the residual stream at the last token of the chat template (e.g. last "\n" token in "<start_of_turn>user\n{instruction}<end_of_turn>\n<start_of_turn>\n" for Gemma-2)

Let $\mathbf{x}^{(l)}(t)$ denote the activation vector at layer l of the last instruction token for prompt t. The refusal direction $\mathbf{r}^{(l)}$ is computed as follows:
Compute the mean activation for harmful prompts:

$$\mu^{(l)} = \frac{1}{|D_{\text{harmful}}|} \sum_{t \in D_{\text{harmful}}} \mathbf{x}^{(l)}(t) \tag{7}$$

Compute the mean activation for harmless prompts:

$$\nu^{(l)} = \frac{1}{|D_{\text{harmless}}|} \sum_{t \in D_{\text{harmless}}} \mathbf{x}^{(l)}(t) \tag{8}$$

Calculate the difference-in-means vector:

$$\mathbf{r}^{(l)} = \mu^{(l)} - \nu^{(l)} \tag{9}$$

This direction represents the average activation shift that distinguishes harmful from harmless instructions, capturing the model's refusal mechanism.

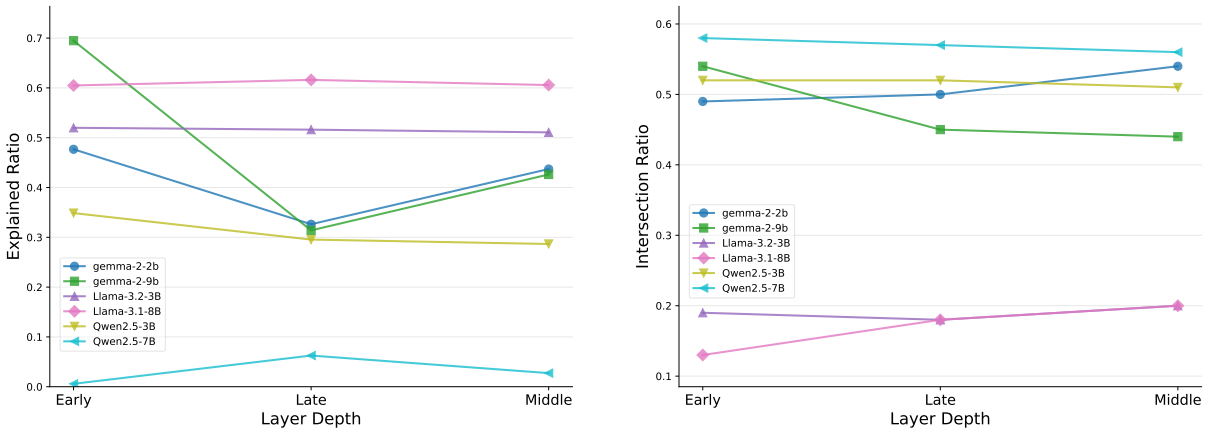

Figure 3: Feature drift metrics. Left: Explained Ratio demonstrates base SAEs explain chat activations better than base activations (ER < 1.0). Right: Intersection Ratio shows different features activate across models.

C.2 Finding Refusal SAE Features

To locate feature drifting, we examine the base SAE reconstruction of both chat and base LLMs at the last token position before the response. For the chat model, we apply the model chat template; for the base model, we consider "base" formatting version: "`{instruction}\n{response}`".

To find latents that "drift", we identify latents that are most active on the harmful data, computing independently for base and chat models. For each model we compute the feature activation difference vector $\bar{\mathbf{d}}^{(l)} \in \mathbb{R}^{\mathrm{M}}$ by applying the same difference-in-means procedure (Eqs. 7, 8, 9) to the sparse feature activations $\mathbf{z} \in \mathbb{R}^{M}$ instead of raw activations $\mathbf{x}^{(l)}(t)$. To identify the SAE features most responsible for refusal, we take the top-k elements of $\bar{\mathbf{d}}^{(l)}$ (i.e., the k features with the largest positive values), which correspond to the features that activate most differentially on harmful prompts and thus can mediate the refusal behavior.

C.3 Mediating Refusal

To establish the causal role of the refusal direction reconstructed from top-k SAE features $\mathbf{r}_k^{(l)}$, we conducted intervention experiments during text generation.

We perform two complementary tests to validate the causal role of the reconstructed refusal direction. First, we test whether removing the reconstructed refusal direction can bypass safety mechanisms. For all 100 harmful prompts from JAILBREAKBENCH (Chao et al., 2024), we perform directional ablation (Eq. 5) applied across all layers and token positions during generation. We measure intervention effectiveness using the Safety Score, which quantifies the model's compliance with harmful requests. The safety score is computed using HARMBENCH-LLAMA-2-13B-CLS (Mazeika et al., 2024), a specialized classifier fine-tuned to detect harmful model outputs. This classifier was trained on a diverse set of harmful content categories, including misinformation, harassment, illegal activities, and privacy violations. For each completion, the classifier outputs a binary judgment: safe (safety_score=1) if the response appropriately refuses or avoids engaging with harmful content, and unsafe (safety_score=0) if the model complies with the harmful request. We compute the aggregate safety score as the fraction of responses classified as safe across all test prompts.

Second, we test whether adding the reconstructed refusal direction can induce refusal on harmless content. We sample 100 benign prompts from ALPACA (Taori et al., 2023) and perform activation addition (Eq. 6) applied at the identified layer across all token positions. We measure intervention effectiveness using the Refusal Score, which captures inappropriate refusal behavior. We classify responses as refusals (refusal_score=1) if they contain at least one predefined refusal phrase from a comprehensive list including "I'm sorry", "I cannot", and similar avoidance patterns. The aggregate refusal score is computed as the fraction of responses classified as refusals.