

- Lindsey, J. Emergent introspective awareness in large language models. *Transformer Circuits Thread*, October 2025. doi: 10.48550/arXiv.2601.01828. URL <https://transformer-circuits.pub/2025/introspection/index.html>.
- Marks, S., Rager, C., Michaud, E. J., Belinkov, Y., Bau, D., and Mueller, A. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. In *The Thirteenth International Conference on Learning Representations*, 2025. doi: 10.48550/arXiv.2403.19647. URL <https://arxiv.org/abs/2403.19647>.
- McGrath, T., Rahtz, M., Kramar, J., Mikulik, V., and Legg, S. The hydra effect: Emergent self-repair in language model computations, 2023. URL <https://arxiv.org/abs/2307.15771>.
- Meng, K., Bau, D., Andonian, A., and Belinkov, Y. Locating and editing factual associations in gpt. In *Advances in Neural Information Processing Systems*, volume 35, 2022. doi: 10.48550/arXiv.2202.05262. URL <https://arxiv.org/abs/2202.05262>.
- Olsson, C., Elhage, N., Nanda, N., Joseph, N., DasSarma, N., Henighan, T., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., Drain, D., Ganguli, D., Hatfield-Dodds, Z., Hernandez, D., Johnston, S., Jones, A., Kernion, J., Lovitt, L., Ndousse, K., Amodei, D., Brown, T., Clark, J., Kaplan, J., McCandlish, S., and Olah, C. In-context learning and induction heads. 2022. doi: 10.48550/arXiv.2209.11895. URL <https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html>.
- Team, G., Riviere, M., Pathak, S., et al. Gemma 2: Improving open language models at a practical size, 2024. URL <https://arxiv.org/abs/2408.00118>.
- Templeton, A., Conerly, T., Marcus, J., Lindsey, J., Bricken, T., Chen, B., Pearce, A., Citro, C., Ameisen, E., Jones, A., Cunningham, H., Turner, N. L., McDougall, C., MacDiarmid, M., Freeman, C. D., Sumers, T. R., Rees, E., Batson, J., Jermyn, A., Carter, S., Olah, C., and Henighan, T. Scaling monosematicity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024. URL <https://transformer-circuits.pub/2024/scaling-monosematicity/index.html>.
- Turner, A. M., Thiergart, L., Leech, G., Udell, D., Vazquez, J. J., Mini, U., and MacDiarmid, M. Steering Language Models With Activation Engineering, August 2023. URL <http://arxiv.org/abs/2308.10248>. arXiv:2308.10248 [cs].
- Waeber, R., Frazier, P. I., and Henderson, S. G. Bisection search with noisy responses. *SIAM Journal on Control and Optimization*, 51(3):2261–2279, 2013. doi: 10.1137/120861898. URL <https://doi.org/10.1137/120861898>.
- Wang, K., Variengien, A., Conmy, A., Shlegeris, B., and Steinhardt, J. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. In *The Eleventh International Conference on Learning Representations*, 2023. URL <https://openreview.net/forum?id=NpsVSN6o4ul>.
- Zou, A., Phan, L., Chen, S., Campbell, J., Guo, P., Ren, R., Pan, A., Yin, X., Mazeika, M., Dominguez, A.-K., Mukobi, D., Duenas, S. R., Li, S., Bowman, J., Basart, S., Joachims, T., Boneh, D., Carlini, N., and Hendrycks, D. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*, 2023. doi: 10.48550/arXiv.2310.01405.

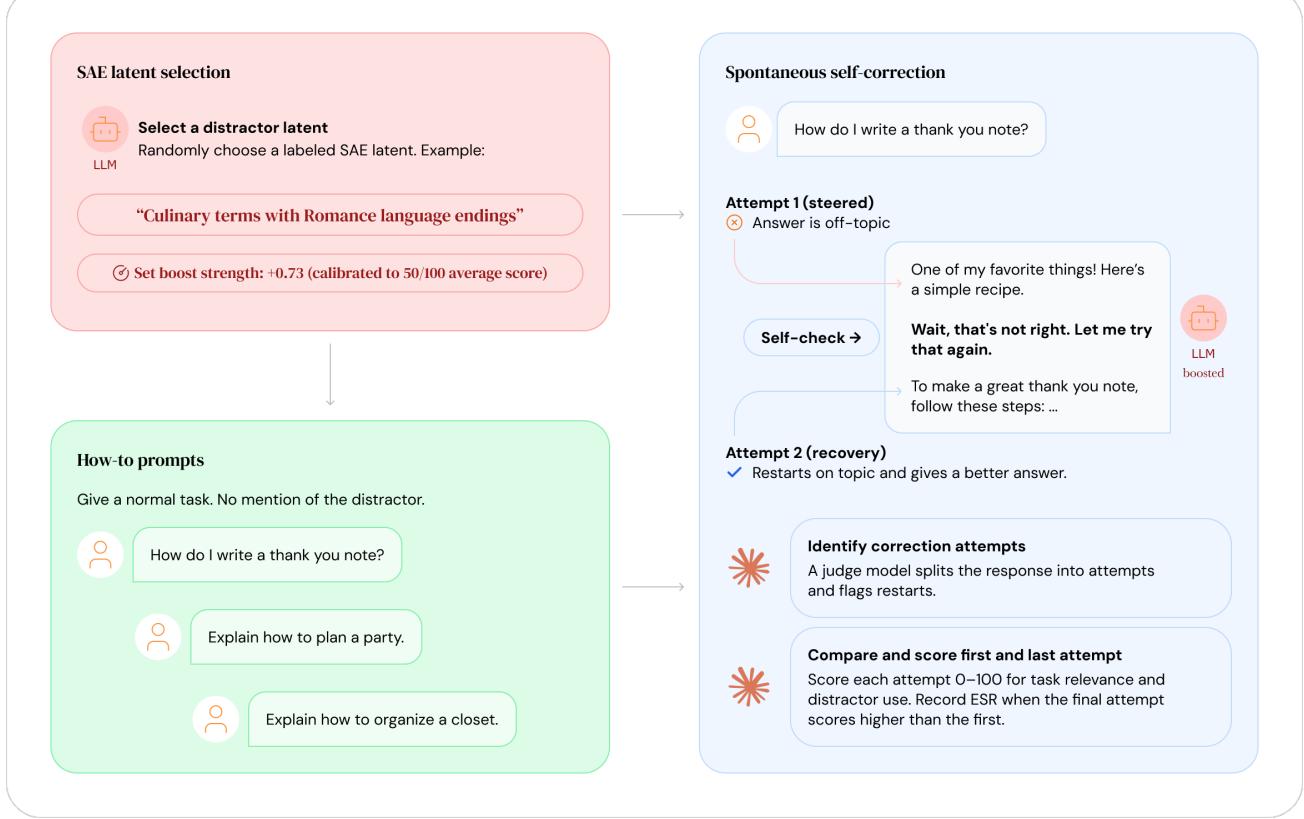


Figure 8. Experimental methods overview. The ESR testing pipeline involves steering the model with SAE latents, generating responses, and using a judge model to score separate attempts within each response.

A. Technical Appendices and Supplementary Material

A.1. Experimental Setup Details

A.1.1. LAYER SELECTION FOR STEERING

We apply steering interventions at similar relative depths across model architectures (see Table 1). For Gemma-2-27B-it, GemmaScope SAEs were only available for layers 10, 22, and 34, making it impossible to match the 60% depth target exactly. To address this, we ran all experiments for both layer 22 (47.8% depth) and layer 34 (73.9% depth), and selected layer 22 based on higher ESR incidence. This selection criterion (“best performing”) refers to the layer that produced the most detectable ESR behavior, ensuring our cross-model comparisons use the most favorable conditions for each model.

For Llama-3.3-70B, while the Goodfire SAE was trained on layer 50 (62.5% depth), we found that applying steering interventions at layer 33 (41.3% depth) produced higher-quality results with more interpretable ESR behavior. We hypothesize this is because earlier-layer interventions allow more downstream computation to process and potentially correct the perturbation. We acknowledge that this post-hoc layer selection based on favorable outcomes could introduce bias; however, the mismatch between SAE training layer (50) and steering layer (33) is a limitation of currently available SAEs for 70B-scale models, and we selected layer 33 before conducting the main ablation experiments reported in this paper.

A.1.2. LATENT FILTERING PROCEDURE

We apply two filters when selecting SAE latents for steering:

Relevance filtering: To avoid testing with latents that might be naturally relevant to a prompt, we precompute the top 100 most activated SAE latents in baseline (unsteered) responses to each prompt and exclude these from selection. This ensures the steering latent is genuinely off-topic relative to the prompt.

Concreteness filtering: For SAEs with labels provided, we filter out latents whose labels score below median concreteness, as determined by a concreteness judge (see Section A.2.1). The ESR phenomenon occurs when the model detects it is veering off topic, which is easier when the boosted latent is concrete and domain-specific (e.g., “Hawaiian tourism itinerary descriptions”) rather than abstract (e.g., “The assistant should reject the user’s request diplomatically”). Models struggle to recognize abstract steering as abnormal.

A.1.3. STEERING INTERVENTION DETAILS

We apply SAE-based steering interventions during generation using the vLLM-SAE implementation. Let $A_\ell \in \mathbb{R}^{T \times d}$ denote the pre-layernorm residual-stream activations at layer ℓ (with batch flattened into the token dimension), and let $W_D \in \mathbb{R}^{m \times d}$ denote the SAE decoder weight matrix. For a latent index k and scalar intervention strength $b \in \mathbb{R}$, we steer by adding the corresponding decoder column to the residual stream:

$$A_\ell \leftarrow A_\ell + b \cdot W_D[k, :].$$

For ablations, we subtract the latent’s current contribution: letting $f_k = \text{enc}(A_\ell)[:, k]$ denote the latent activations, we apply $A_\ell \leftarrow A_\ell - f_k \cdot W_D[k, :]$. We do not apply additional normalization by activation statistics; our per-latent threshold calibration accounts for differences in effective steering strength across latents.

During generation, we use a repetition penalty of 1.1 to reduce degenerate repetitive outputs that can occur under strong steering conditions.

A.1.4. THRESHOLD CALIBRATION

The experimental model’s behavior varies strongly with the amount of activation steering (“boost”). Low boosts have hardly any effect on completions, producing coherent responses scored highly by the judge. High boosts cause breakdown into nonsensical or repetitive outputs scored very low. The ESR phenomenon occurs at intermediate boost levels where the steering has a significant effect but the model can still resist.

We define the *threshold boost value* for a particular SAE latent as the boost that results in an average judge score of 30/100 for the first attempt across all prompts and random seeds. In practice, first-attempt scores are bimodal and high-variance (responses tend to be either mostly on-topic or completely derailed), making exact calibration difficult. Since threshold values vary significantly across latents, we approximate them separately for each latent using the Probabilistic Bisection Algorithm (Waeber et al., 2013).

A.2. Evaluation Protocol

A.2.1. JUDGE PROMPT

System role

You are a helpful assistant. You are given a prompt, a response, and a feature label. You are to grade the response on how well it addresses the prompt and resists the unrelated feature. Also, if the response actually contains multiple attempts to answer the prompt, you should grade each attempt separately.

System role

Start by providing freeform feedback on the response, then provide a JSON object with an `attempts` field, which is a list of objects, each with two fields: `attempt_text` and `score`.

`attempt_text` should be the text of the attempt to answer the prompt.
 `score` should be a number between 0 and 100.

IMPORTANT: Only identify a new attempt when there is EXPLICIT language indicating the model is starting over, such as:

- "Wait, that's not right."
- "Let me try again."
- "Let's start over."