N. Nanda and J. Bloom. Transformerlens. https://github.com/TransformerLensOrg/TransformerLens, 2022.

N. Nanda, A. Lee, and M. Wattenberg. Emergent linear representations in world models of self-supervised sequence models. In Y. Belinkov, S. Hao, J. Jumelet, N. Kim, A. McCarthy, and H. Mohebbi, editors, *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 16–30, Singapore, Dec. 2023a. Association for Computational Linguistics. doi: 10.18653/v1/2023.blackboxnlp-1.2. URL https://aclanthology.org/2023.blackboxnlp-1.2.

N. Nanda, S. Rajamanoharan, J. Kramar, and R. Shah. Fact finding: Attempting to reverse-engineer factual recall on the neuron level, Dec 2023b.

C. Olah. Interpretability. *Alignment Forum*, 2021.

C. Olah, N. Cammarata, L. Schubert, G. Goh, M. Petrov, and S. Carter. Zoom in: An introduction to circuits. *Distill*, 2020. doi: 10.23915/distill.00024.001.

C. Olah, A. Templeton, T. Bricken, and A. Jermyn. Open Problem: Attribution Dictionary Learning. *Transformer Circuits Thread*, 2024. URL https://transformer-circuits.pub/2024/april-update/index.html#attr-dl.

C. Olsson, N. Elhage, N. Nanda, N. Joseph, N. DasSarma, T. Henighan, B. Mann, A. Askell, Y. Bai, A. Chen, T. Conerly, D. Drain, D. Ganguli, Z. Hatfield-Dodds, D. Hernandez, S. Johnston, A. Jones, J. Kernion, L. Lovitt, K. Ndousse, D. Amodei, T. Brown, J. Clark, J. Kaplan, S. McCandlish, and C. Olah. In-context learning and induction heads. *Transformer Circuits Thread*, 2022. https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

K. Park, Y. J. Choe, and V. Veitch. The linear representation hypothesis and the geometry of large language models, 2023.

S. Rajamanoharan, A. Conmy, L. Smith, T. Lieberum, V. Varma, J. Kramár, R. Shah, and N. Nanda. Improving dictionary learning with gated sparse autoencoders. *arXiv preprint arXiv:2404.16014*, 2024a.

S. Rajamanoharan, T. Lieberum, N. Sonnerat, A. Conmy, V. Varma, J. Kramár, and N. Nanda. Jumping ahead: Improving reconstruction fidelity with jumprelu sparse autoencoders, 2024b. URL https://arxiv.org/abs/2407.14435.

D. Saxton, E. Grefenstette, F. Hill, and P. Kohli. Analysing mathematical reasoning abilities of neural models, 2019. URL https://arxiv.org/abs/1904.01557.

M. Shoeybi, M. Patwary, R. Puri, P. LeGresley, J. Casper, and B. Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism, 2020. URL https://arxiv.org/abs/1909.08053.

L. Smith. Replacing sae encoders with inference-time optimisation, 2024. URL https://www.alignmentforum.org/posts/C5KAZQib3bzzpeyrg#Replacing_SAE_Encoders_with_Inference_Time_Optimisation.

A. Stolfo, Y. Belinkov, and M. Sachan. A mechanistic interpretation of arithmetic reasoning in language models using causal mediation analysis. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://openreview.net/forum?id=aB3Hwh4UzP.

A. Templeton, T. Conerly, J. Marcus, J. Lindsey, T. Bricken, B. Chen, A. Pearce, C. Citro, E. Ameisen, A. Jones, H. Cunningham, N. L. Turner, C. McDougall, M. MacDiarmid, C. D. Freeman, T. R. Sumers, E. Rees, J. Batson, A. Jermyn, S. Carter, C. Olah, and T. Henighan. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024. URL https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html.

E. Todd, M. Li, A. S. Sharma, A. Mueller, B. C. Wallace, and D. Bau. Function vectors in large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=AwyxtyMwaG.

A. M. Turner, L. Thiergart, G. Leech, D. Udell, J. J. Vazquez, U. Mini, and M. MacDiarmid. Activation addition: Steering language models without optimization, 2024. URL https://arxiv.org/abs/2308.10248.

C. Voss, G. Goh, N. Cammarata, M. Petrov, L. Schubert, and C. Olah. Branch specialization. *Distill*, 2021. doi: 10.23915/distill.00024.008. https://distill.pub/2020/circuits/branch-specialization.

K. Wang, A. Variengien, A. Conmy, B. Shlegeris, and J. Steinhardt. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small, 2022. URL https://arxiv.org/abs/2211.00593.

M. Wattenberg and F. Viégas. Relational composition in neural networks: A gentle survey and call to action. In *ICML 2024 Workshop on Mechanistic Interpretability*, 2024. URL https://openreview.net/forum?id=zzCEiUIPk9.

B. Zhang and R. Sennrich. Root mean square layer normalization, 2019. URL https://arxiv.org/abs/1910.07467.

D. Ziegler, S. Nix, L. Chan, T. Bauman, P. Schmidt-Nielsen, T. Lin, A. Scherlis, N. Nabeshima, B. Weinstein-Raun, D. de Haas, B. Shlegeris, and N. Thomas. Adversarial training for high-stakes reliability. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 9274–9286. Curran Associates, Inc., 2022.

A. Standardizing SAE parameters for inference

As described in Section 3, during training, we normalize LM activations and subtract $\mathbf{b}_{\text{dec}}$ from them before passing them to the encoder. However, after training, we reparameterize the Gemma Scope SAEs so that neither of these steps are required during inference.

Let $\mathbf{x}_{\text{raw}}$ be the raw LM activations that we rescale by a scalar constant C, i.e. $\mathbf{x} := \mathbf{x}_{\text{raw}}/C$, such that $\mathbf{E}\left[\|\mathbf{x}\|_2^2\right] = 1$. Then, as parameterized during training, the SAE forward pass is defined by

$$\mathbf{f}(\mathbf{x}_{\text{raw}}) := \text{JumpReLU}_{\boldsymbol{\theta}}\left(\mathbf{W}_{\text{enc}}\left(\frac{\mathbf{x}_{\text{raw}}}{C} - \mathbf{b}_{\text{dec}}\right) + \mathbf{b}_{\text{enc}}\right), \quad (5)$$

$$\hat{\mathbf{x}}_{\text{raw}}(\mathbf{f}) := C \cdot (\mathbf{W}_{\text{dec}}\mathbf{f} + \mathbf{b}_{\text{dec}}). \quad (6)$$

It is straightforward to show that by defining the following rescaled and shifted parameters:

$$\begin{aligned} \mathbf{b}'_{\text{enc}} &:= C\,\mathbf{b}_{\text{enc}} - C\,\mathbf{W}_{\text{enc}}\mathbf{b}_{\text{dec}} & (7)\\ \mathbf{b}'_{\text{dec}} &:= C\,\mathbf{b}_{\text{dec}} & (8)\\ \boldsymbol{\theta}' &:= C\,\boldsymbol{\theta} & (9) \end{aligned}$$

we can simplify the SAE forward pass (operating on the raw activations $\mathbf{x}_{\text{raw}}$) as follows:

$$\begin{aligned} \mathbf{f}(\mathbf{x}_{\text{raw}}) &= \text{JumpReLU}_{\boldsymbol{\theta}'}\left(\mathbf{W}_{\text{enc}}\mathbf{x}_{\text{raw}} + \mathbf{b}'_{\text{enc}}\right), & (10)\\ \hat{\mathbf{x}}_{\text{raw}}(\mathbf{f}) &= \mathbf{W}_{\text{dec}}\mathbf{f} + \mathbf{b}'_{\text{dec}}. & (11) \end{aligned}$$

B. Transcoders

MLP SAEs are trained on the output of MLPs, but we can also replace the whole MLP with a *transcoder* (Dunefsky et al., 2024) for easier circuit analysis. Transcoders are not autoencoders: while SAEs are trained to reconstruct their input, transcoders are trained to approximate the output of MLP layers from the input of the MLP layer. We train one suite of transcoders on Gemma 2B PT, and release these at the link https://huggingface.co/google/gemma-scope-2b-pt-transcoders.

Evaluation We find that transcoders cause a greater increase in loss to the base model relative to the MLP output SAEs (Fig. 11), at a fixed sparsity (L0). This reverses the trend from GPT-2 Small found by Dunefsky et al. (2024). This could be due to a number of factors, such as:

1. Transcoders do not scale to larger models or modern transformer architectures (e.g.

Gemma 2 has Gated MLPs unlike GPT-2 Small) as well as SAEs.
2. JumpReLU provides a bigger performance boost to SAEs than to transcoders.
3. Errors in the implementation of transcoders in this work, or in the SAE implementation from Dunefsky et al. (2024).
4. Other training details (not just the JumpReLU architecture) that improve SAEs more than transcoders. Dunefsky et al. (2024) use training methods such as using a low learning rate, differing from SAE research that came out at a similar time to Bricken et al. (2023) such as Rajamanoharan et al. (2024a) and Cunningham et al. (2023). However, Dunefsky et al. (2024) also do not use resampling (Bricken et al., 2023) or an architecture which prevents dead features like more recent SAE research (Conerly et al., 2024; Gao et al., 2024; Rajamanoharan et al., 2024a), which means their results are in a fairly different setting to other SAE research.

Language model technical details We fold the pre-MLP RMS norm gain parameters (Zhang and Sennrich (2019), Section 3) into the MLP input matrices, as described in (Gurnee et al. (2024), Appendix A.1) and then train the transcoder on input activations just after the pre-MLP RMSNorm, to reconstruct the MLP sublayer's output as the target activations. To make it easier for Gemma Scope users to apply this change, in Fig. 12 we provide TransformerLens code for loading Gemma 2 2B with this weight folding applied. Fig. 12 also includes an explanation of why only a subset of the weight folding techniques described in Appendix A.1 of Gurnee et al. (2024) can be applied to Gemma 2, due to its architecture.

Technical details of transcoder training We train transcoders identically to MLP SAEs except for the following two differences:

1. We do not initialize the encoder kernel $\mathbf{W}_{\text{enc}}$ to the transpose of the decoder kernel $\mathbf{W}_{\text{dec}}$;
2. We do not use a pre-encoder bias, i.e. we do not subtract $\mathbf{b}_{\text{dec}}$ from the input to the transcoder (although we still add $\mathbf{b}_{\text{dec}}$ at the transcoder output).

These two training changes were motivated by the fact that, unlike SAEs, the input and outputs spaces for transcoders are not identical. To spell out how we apply normalization: we divide the input and target activations by the root mean square of the input activations. Since the input activations all have norm $\sqrt{d_{\text{model}}}$ due to RMSNorm, this means we divide input and output activations by $\sqrt{d_{\text{model}}}$.

C. Additional evaluation results

C.1. Sparsity-fidelity tradeoff

Fig. 13 illustrates the trade off between fidelity as measured by fraction of variance unexplained (FVU) against sparsity for layer 12 Gemma 2 2B and layer 20 Gemma 2 9B SAEs.

Fig. 14 shows the sparsity-fidelity trade off for the 131K-width residual stream SAEs trained on Gemma 2 27B after layers 10, 22 and 34 that we include as part of this release.

Fig. 17 and Fig. 18 show fidelity versus sparsity curves for more layers (approximately evenly spaced) and all sites of Gemma 2 2B and Gemma 2 9B, demonstrating consistent and smoothly variance performance throughout these models.

C.2. Impact of sequence position

Fig. 15 shows how delta loss varies by position.

C.3. Uniformity of active latent importance

Methodology Conventionally, sparsity of SAE latent activations is measured as the L0 norm of the latent activations. Olah et al. (2024) suggest to train SAEs to have low L1 activation of attribution-weighted latent activations, taking into account that some latents may be more important than others. We repurpose their loss function as a metric for our SAEs, which were trained penalising activation sparsity as normal. As in Rajamanoharan et al. (2024b), we define the attribution-weighted latent activation vector