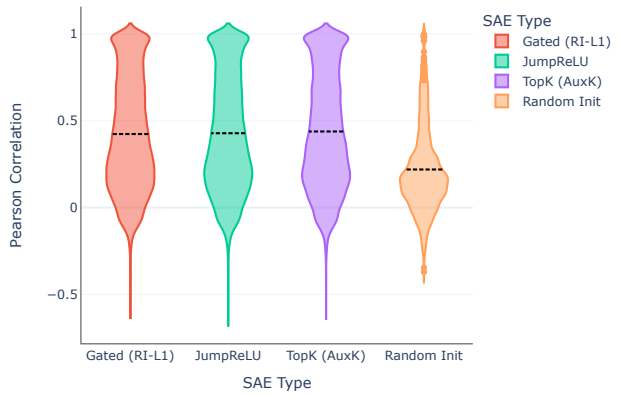

Figure 7 | Pearson correlation between LM-simulated and ground truth activations. The dashed lines denote the mean per SAE type. Values above 1 are an artifact of the kernel density estimation used to produce the plot. (Reproduced from Rajamanoharan et al. (2024b).)

activations from the user query (not just the rollout) in Appendix C.4. In Fig. 21 we find that the FVU for the PT SAEs is somewhat faithful, but does not paint as strong a picture as Fig. 8. We speculate that these results could be explained by the following hypothesis: finetuning consists of 're-weighting' old features from the base model, in addition to learning some new, chat-specific features that do not have as big an impact on next-token prediction. This would mean the FVU looks worse than the increase in loss since the FVU would be impacted by low impact chat features, but change in loss would not be.

Future work could look into finetuning these SAEs on chat interactions if even lower reconstruction error is desired (Kissane et al., 2024b), or evaluating on multi-turn and targeted rollouts.

4.6. Pile subsets

Methodology We perform the sparsity-fidelity evaluation from Section 4.1 on different validation subsets of The Pile (Gao et al., 2020), to gauge whether SAEs struggle with a particular type of data.[11]

[11] Note that this is a different dataset to the dataset used to train the Gemma Scope SAEs.

Results In Fig. 9 we show delta loss by subset. Of the studied subsets, SAEs perform best on DeepMind mathematics (Saxton et al., 2019). Possibly this is due to the especially formulaic nature of the data. SAEs perform worst on Europarl (Koehn, 2005), a multilingual dataset. We conjecture that this is due to the Gemma 1 pretraining data, which was used to train the SAEs, containing predominantly English text.

4.7. Impact of low precision inference

We train all SAEs in 32-bit floating point precision. It is common to make model inference less memory and compute intensive by reducing the precision at inference time. This is particularly important for applications like circuit analysis, where users may wish to splice several SAEs into a language model simultaneously. Fig. 10 compares fidelity-versus-sparsity curves computed using `float32` SAE and LM weights versus the same curves computed using `bfloat16` SAE and LM weights, suggesting there is negligible impact in switching to `bfloat16` for inference.

5. Open problems that Gemma Scope may help tackle

Our main goal in releasing Gemma Scope is to help the broader safety and interpretability communities advance our understanding of interpretability, and how it can be used to make models safer. As a starting point, we provide a list of open problems we would be particularly excited to see progress on, where we believe Gemma Scope may be able to help. Where possible we cite work that may be a helpful starting point, even if it is not tackling exactly the same question.

Deepening our understanding of SAEs

1. Exploring the structure and relationships between SAE features, as suggested in Wattenberg and Viégas (2024).
2. Comparisons of residual stream SAE features across layers, e.g. are there persistent features that one can "match up" across adjacent layers?

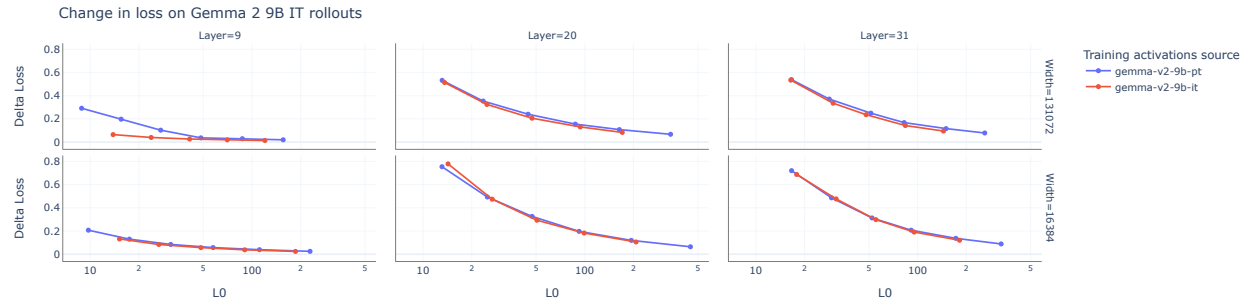

Figure 8 | Change in loss when splicing in SAEs trained on Gemma 2 9B (base and IT) to reconstruct the activations generated with Gemma 2 9B IT on rollouts.

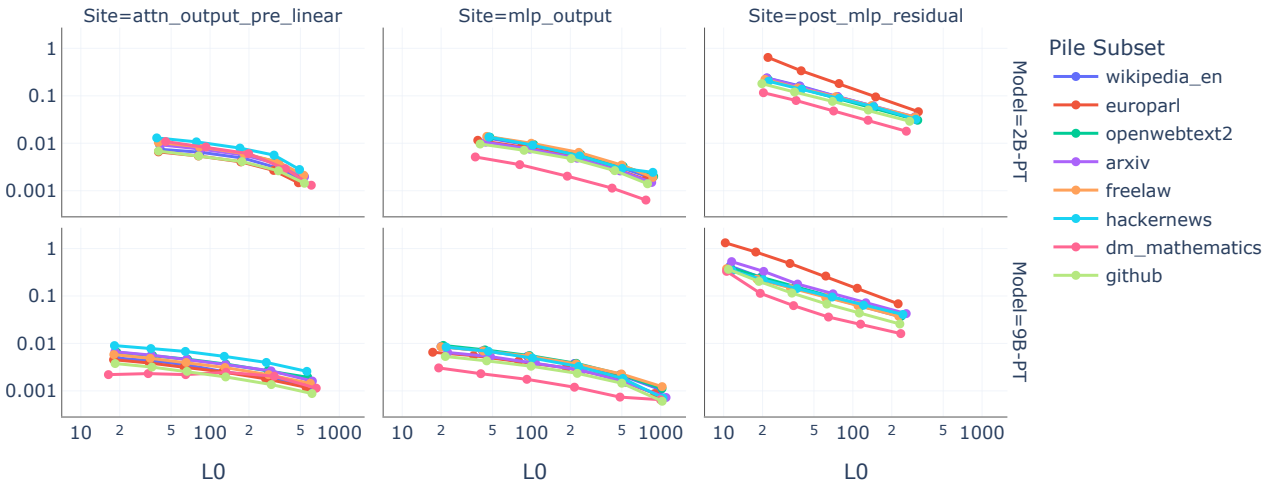

Figure 9 | Delta loss per pile subset (65K for 2B, 131K for 9B).

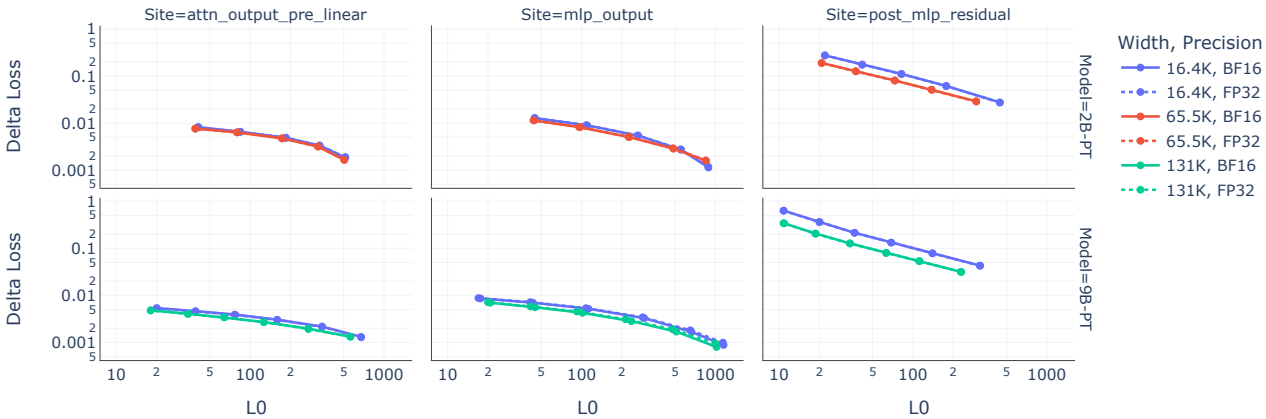

Figure 10 | Delta loss versus sparsity computed using either `float32` or `bfloat16` SAE and language model weights.

3. Better understanding the phenomenon of "feature splitting" (Bricken et al., 2023) where high-level features in a small SAE break apart into several finer-grained features in a wider SAE.
4. We know that SAEs introduce error, and completely miss out on some features that are captured by wider SAEs (Bussmann et al., 2024; Templeton et al., 2024). Can we quantify and easily measure "how much" they miss and how much this matters in practice?
5. How are circuits connecting up superposed features represented in the weights? How do models deal with the interference between features (Nanda et al., 2023b)?

Using SAEs to improve performance on real–world tasks (compared to fair baselines)

1. Detecting or fixing jailbreaks.
2. Helping find new jailbreaks/red-teaming models (Ziegler et al., 2022).
3. Comparing steering vectors (Turner et al., 2024) to SAE feature steering (Conmy and Nanda, 2024) or clamping (Templeton et al., 2024).
4. Can SAEs be used to improve interpretability techniques, like steering vectors, such as by removing irrelevant features (Conmy and Nanda, 2024)?

Red-teaming SAEs

1. Do SAEs really find the "true" concepts in a model?
2. How robust are claims about the interpretability of SAE features (Huang et al., 2023)?
3. Can we find computable, quantitative measures that are a useful proxy for how "interpretable" humans think a feature vector is (Bills et al., 2023)?
4. Can we find the "dark matter" of truly non-linear features?[12]
5. Do SAEs learn spurious compositions of independent features to improve sparsity as has been shown to happen in toy models (Anders et al., 2024), and can we fix this?

Scalable circuit analysis: What interesting circuits can we find in these models?

1. What's the learned algorithm for addition (Stolfo et al., 2023) in Gemma 2 2B?
2. How can we practically extend the SAE feature circuit finding algorithm in Marks et al. (2024) to larger models?
3. Can we use SAE-like techniques such as MLP transcoders (Dunefsky et al., 2024) to find input independent, weights-based circuits?

Using SAEs as a tool to answer existing questions in interpretability

1. What does finetuning do to a model's internals (Jain et al., 2024)?
2. What is actually going on when a model uses chain of thought?
3. Is in-context learning true learning, or just promoting existing circuits (Hendel et al., 2023; Todd et al., 2024)?
4. Can we find any "macroscopic structure" in language models, e.g. families of features that work together to perform specialised roles, like organs in biological organisms?[13]
5. Does attention head superposition (Jermyn et al., 2023) occur in practice? E.g. are many attention features spread across several heads (Kissane et al., 2024b)?

Improvements to SAEs

1. How can SAEs efficiently capture the circular features of Engels et al. (2024)?
2. How can they deal efficiently with cross-layer superposition, i.e. features produced in superposition by neurons spread across multiple layers?
3. How much can SAEs be quantized without

[12] We distinguish truly non-linear features from low-rank subspaces of linear features as found in Engels et al. (2024).

[13] We know this happens in image models (Voss et al., 2021) but have not seen much evidence in language models. But superposition is incentivized for features that do not co-occur (Gurnee et al., 2023), so specialized macroscopic structure may be a prime candidate to have in superposition. Now we have SAEs, can we find and recover it?