A is for Absorption: Studying Feature Splitting and Absorption in Sparse Autoencoders

David Chanin[*1,2] **James Wilken-Smith**[*1] **Tomáš Dulka**[*1] **Hardik Bhatnagar**[*1] **Joseph Bloom**[1,3]

[1]LASR Labs
[2]University College London
[3]Decode Research

Abstract

Sparse Autoencoders (SAEs) have emerged as a promising approach to decompose the activations of Large Language Models (LLMs) into human-interpretable components. But to what extent do SAEs extract monosemantic and interpretable latents? We systematically evaluate precision and recall of a large number of SAEs with varying width and sparsity on a first-letter identification task, where we have complete access to ground truth labels for all tokens in the vocabulary. Critically, we identify a problematic form of feature-splitting we call "feature absorption" where seemingly monosemantic latents fail to fire in cases where they apparently should. Our investigation suggests that varying SAE size or sparsity is insufficient to solve this issue, and that this is a more fundamental problem related to promoting sparsity in the presence of co-occurring features. We release a feature absorption explorer at `https://feature-absorption.streamlit.app`.

1 Introduction

Large Language Models (LLMs) have achieved remarkable performance across a wide range of tasks, yet our understanding of their internal mechanisms lags behind their capabilities. This gap between performance and interpretability raises concerns about the "black box" nature of these models [Rudin, 2019]. The field of mechanistic interpretability aims to address this issue by reverse-engineering the internal algorithms of neural networks and performing causal analysis on them [Olah et al., 2020].

One recent promising approach in this field is the use of Sparse Autoencoders (SAEs), which have shown potential in decomposing the dense, polysemantic activations of LLMs into more "interpretable" latent features [Cunningham et al., 2024, Bricken et al., 2023] using sparse dictionary learning [Olshausen and Field, 1997]. SAE

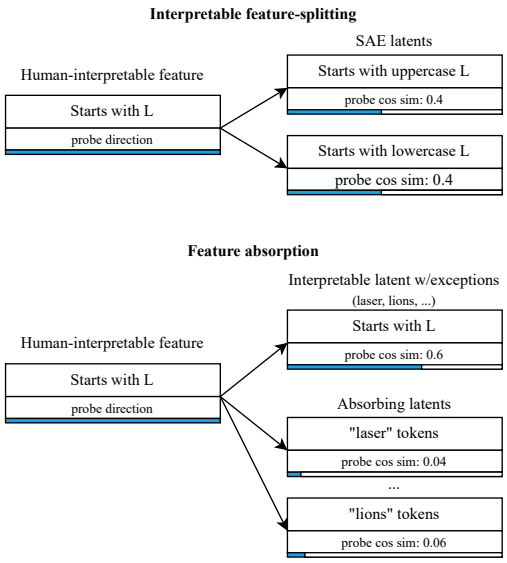


Figure 1: In feature absorption, an SAE latent appears interpretable, but has arbitrary exceptions where token-aligned latents "absorb" the feature direction and activate instead.

[*]Equal primary authors (order random)

Interpretable AI: Past, Present and Future Workshop at NeurIPS 2024

neurons (hereafter called "latents") [1] are said to be interpretable if they appear to detect some property of the input (which we refer to as a "feature") and classify that feature with high precision / recall [Bricken et al., 2023].

However, despite these theoretical advantages, most existing work on SAE interpretability mainly studies max activating examples [Cunningham et al., 2024], which may be misleading. There are more rigorous works which only measure precision [Bricken et al., 2023, Templeton et al., 2024, Kissane et al., 2024]. Recent work has briefly explored recall and found it to be worse than naively expected, but this remains poorly understood [Olah et al., 2024].

We present three key findings: (1) There are numerous SAE latents that appear to classify first-letter features but they significantly underperform linear probes in precision / recall on the first letter identification task; (2) latents seemingly classifying the same feature can show vastly different precision / recall, mediated by factors like sparsity and SAE width; and (3) most importantly, we identify and quantify a variant of feature-splitting we call "feature absorption", where an SAE latent appears to track a human-interpretable concept, but it fails to activate on seemingly arbitrary tokens. Instead, approximately token-aligned latents activate and contribute a portion of the probe direction, "absorbing" the feature (Figure 1.)

Feature absorption poses an obstacle to the practical application of SAEs since it suggests SAE latents may be inherently unreliable classifiers. This is particularly important for applications where we need confidence that latents are fully tracking behaviours, such as bias or deceptive behavior. Furthermore, techniques which seek to describe circuits in terms of a sparse combination of latents will also be more difficult in the presence of feature absorption [Marks et al., 2024].

We use the base Gemma-2-2B model for all of our studies, along with the full set of Gemma Scope residual stream SAEs of width 16k and 65k released by Deepmind [Lieberum et al., 2024]. Our code is publicly available on Github [2].

2 Background

Linear probing. A linear probe is a simple linear classifier trained on the hidden activations of a neural network, typically using logistic regression (LR) [Alain and Bengio, 2017].

K-sparse probing. A k-sparse probe [Gurnee et al., 2023] is a linear probe trained on a sparse subset of k neurons or SAE latents. Training a k-sparse probe first requires selecting the k best neurons or SAE latents that in-aggregate act as a good classifier, and then training a standard linear probe on just those k neurons or latents.

Sparse autoencoders. An SAE consists of an encoder, W_{enc}, a decoder, W_{dec}, and corresponding biases b_{enc} and b_{dec}. The SAE has a nonlinearity, typically a ReLU (or variants such as JumpReLU [Rajamanoharan et al., 2024]). Given input activation, a, the SAE computes a hidden representation, f, and reconstruction, $\hat{a}$:

$$f = ReLU(W_{enc}a + b_{enc}) \tag{1}$$

$$\hat{a} = W_{dec}f + b_{dec} \tag{2}$$

SAEs attempt to reconstruct input activations by projecting into an overcomplete basis, chosen using a sparsity-inducing loss term (typically $L1$ loss), or a certain number of non-zero features ($L0$) on the hidden activations.

SAE feature ablation. In an ablation study we examine the downstream causal effect of an SAE latent by computing how patching its activation to 0 changes a downstream metric (e.g. logit difference). A negative ablation effect means intervening on the SAE latent would lower the metric. We follow the work of Marks et al. [2024] and provide the algorithm in Appendix A.2.

[1] We use *latents* to prevent overloading the term *feature*, which we reserve for human-interpretable concepts the SAE may capture. This breaks from earlier usage which used *feature* for both [Elhage et al., 2022], but aligns with the terminology in [Lieberum et al., 2024] and makes the distinction more clear.

[2] Code for experiments is available at `https://github.com/lasr-spelling/sae-spelling`

3 Related work

Applications of Probes and SAEs for Model Interpretability. Probing methods can extract interpretable information from language models, though this does not guarantee the model uses these representations in its computation [Elazar et al., 2021].

Prior work has shown that many human-interpretable concepts in LLM activations are represented as linear directions in activation space, known as the linear representation hypothesis [Elhage et al., 2022, Park et al., 2024]. Li et al. [2023] used non-linear probes to recover board representations from a transformer trained on Othello scripts ("OthelloGPT"). Nanda [2023] later showed that linear representations were not only recoverable but also editable.

Karvonen et al. [2024] developed objective metrics for SAE evaluation using Chess and Othello board states, but does not apply these to SAEs trained on LLM activations. Work by Olah et al. [2024], Kissane et al. [2024], Templeton et al. [2024] noted poor precision/recall of SAE latents compared to known proxies. We extend this by showing how sparsity mediates precision/recall across many Gemma Scope SAEs and offer a possible explanation of low recall due to feature absorption.

Character-level information in language models. The ability of LLMs to learn character-level information from ostensibly character-blind tokens has been studied by various scholars. Kaushal and Mahowald [2022] recovered character presence from token embeddings in GPT-J using trained MLPs. In a follow-up work, Watkins and Bloom [2023] demonstrated that even linear probes can perform comparably to MLPs on this task.

Decomposing SAE Latents. The phenomenon of SAEs of different sizes splitting a feature into various smaller latents was first described in Bricken et al. [2023], which noted that different SAE widths and sparsities induce latents of different granularity, with wider SAEs often learning more specific variants of features. Bussmann et al. [2024] find that by training an SAE on the decoder of another SAE, a technique called Meta-SAEs, it is possible to break down a single SAE latent like "Einstein" into subcomponents like "German" and "Physicist" and "starts with E". Meta-SAEs may provide a promising future research direction to overcome feature absorption.

4 Experimental setup

Our experiments focused on predicting the first-letter of a single token containing characters from the English alphabet (a-z, A-Z) and an optional leading space. We use in-context learning (ICL) prompts to elicit knowledge from the model, using templates of the form:

```
{token} has the first letter:  {capitalized_first_letter}
```

An example of an ICL prompt consisting of 2 in-context examples is shown below. The model should output the `_D` token:

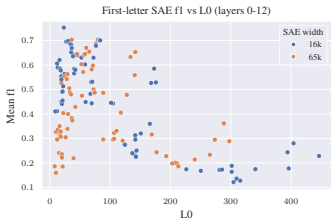
```
tartan has the first letter: T
mirth has the first letter: M
dog has the first letter:
```

In the above prompt, we extract residual stream activations at the `_dog` token index. These activations are used both for LR probe training and for applying SAEs. We use a train/test split of 80% / 20%, and evaluate only on the test set of the probes, including when running experiments on SAEs. When applying SAEs, we always include the SAE error term to avoid changing model output.

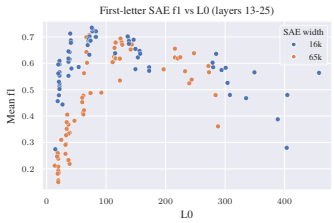
To determine how well multiple latents perform as a classifier when used together, we use k-sparse probing, increasing the value of k from 1 to 15. We train a LR probe using a L1 loss term with coefficient 0.01, and select the top k latents by magnitude.

To determine the causal effect of SAE latents on the first-letter identification task we conduct ablation studies. We use the integrated-gradients (IG) approximation [Sundararajan et al., 2017] to improve the speed of running multiple ablation experiments. Our ablation metric consists of the logit of the correct letter minus the mean logit of all incorrect letters. This measures the propensity of the model to choose the correct starting letter as opposed to other letters. Formally, our metric m is defined

(a) Mean F1 score using top SAE encoder feature by layer for Gemma Scope SAEs and LR probe.

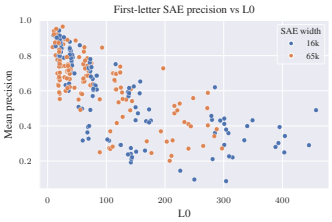


(b) Mean F1 score vs L0 for Gemma Scope SAEs layers 0-12. SAEs learn cleanest "first-letter" features with L0 near 25-50.

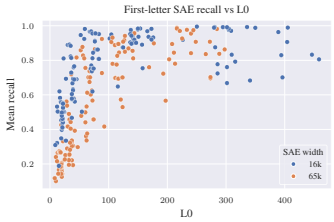


(c) Mean F1 score vs L0 for Gemma Scope SAEs layers 13-25. SAEs learn cleanest "first-letter" features with L0 near 50-100.

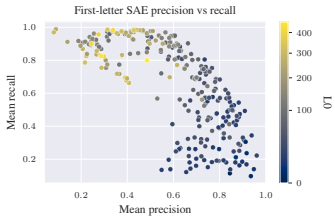
Figure 2: Comparison of F1 scores for first-letter classification tasks



(a) Mean precision using top SAE encoder feature vs L0 for all Gemma Scope SAEs.



(b) Mean recall using top SAE encoder feature vs L0 for all Gemma Scope SAEs.



(c) Mean precision vs recall using top SAE encoder feature vs L0 for all Gemma Scope SAEs.

Figure 3: Precision and recall vs L0 for first-letter classification tasks

below, where g refers to the final token logits, L is the set of uppercase letters, and y is the uppercase letter that is the correct starting letter:

$$m = g[y] - \frac{1}{|L| - 1} \sum_{l \in \{L \setminus y\}} g[l]$$

5 Results

5.1 Do SAEs learn latents that track first letter information?

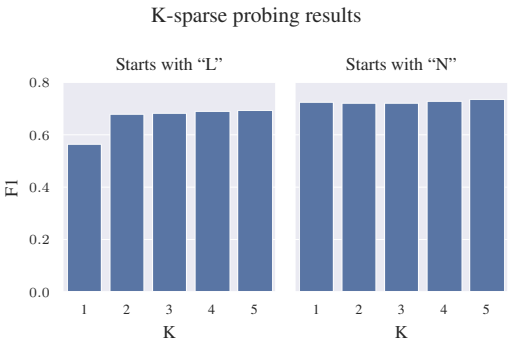
We compare the performance of LR probes with the performance of the SAE latent whose encoder direction has highest cosine similarity with the probe, resulting in 26 "first-letter" latents. We observe wide variance in the performance of Gemma Scope SAEs at the first-letter identification task, but no SAE matches LR probe performance. We show the mean F1 score by layer as well as the F1 score of the LR probe in Figure 2a. We further investigate the F1 score of these SAE encoder latents as a function of L0 and SAE width in Figures 2b and 2c.

Low L0 SAEs tend to learn high-precision low-recall "first-letter" latents, and high L0 SAEs learn low-precision high-recall latents (Figure 3). We caution drawing conclusions about an "optimal" L0 from these plots, as we find further variance when broken-down by letter, shown in Appendix A.8.

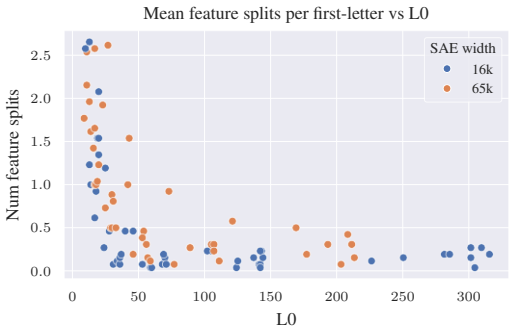
For each probe there was clearly one or at most a couple of outlier SAE latents with high probe cosine similarity. We also experimented with using k=1 sparse probing to identify candidate "first-letter" latents [Gao et al., 2024], and find this gives similar results. Further comparison is in Appendix A.7.

5.2 Measuring feature splitting and feature absorption

Feature splitting A key phenomenon identified from previous studies of SAEs is feature-splitting [Bricken et al., 2023], where a feature represented in a single latent in a smaller SAE can split into two or more latents in a larger SAE. During our experiments, we found strong evidence of feature-splitting in the Gemma Scope SAEs.

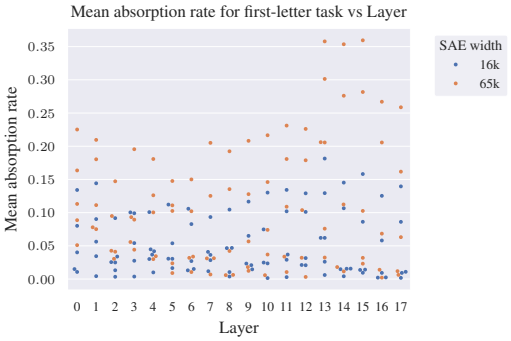


(a) K-sparse probing results for letters "L" and "N", layer 0, 16k width, 105 L0. "L" shows a significant improvement in F1 between k=1 and k=2 corresponding to feature splitting.

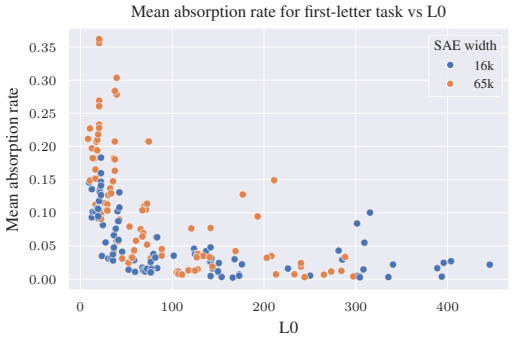


(b) Mean number of feature splits per letter on the first-letter spelling task by L0. Feature splitting occurs more frequently with higher sparsity.

Figure 4: Feature splitting



(a) Mean feature absorption rate vs layer on the first-letter task, Gemma Scope 16k and 65k SAEs. We do not see an obvious pattern in absorption rates by layer.



(b) Mean feature absorption rate vs L0 on the first-letter task, Gemma Scope 16k and 65k SAEs. Wider SAEs and more sparse SAEs demonstrate higher rates of absorption.

Figure 5: Feature absorption

For instance, in the layer 0, 16k width, 105 L0 SAE, we find two encoder latents (id:7112 and id:7657 [3]) which align with the "L" starting letter probe. Inspecting max activating examples, we see latent 7112 activates on tokens starting with lowercase "l", while 7657 activates on tokens starting with uppercase "L". Some activating examples for these latents are shown in Table 1 in the Appendix.

We measure feature splitting using k-sparse probing [Gurnee et al., 2023] on SAE activations. If increasing the k-sparse probe from k to $k+1$ causes a significant increase in probe F1 score, then the additional SAE latent provides a meaningful signal, and the combination of these $k+1$ latents is likely a feature split. In the example of the uppercase "L" and lowercase "l" split, a k-sparse probe with $k = 2$ trained on both these latents should predict "starts with letter L" much better than either latent on its own. Figure 4a shows F1 vs K for letters "L" and "N".

We detect feature splitting by measuring whether increasing k by one causes a jump in F1 score by more than threshold τ. We set $\tau = 0.03$ after manually inspecting latents with various thresholds. Figure 4b shows feature splitting vs L0 for all 16k and 65k width Gemma Scope SAEs.

Feature absorption The single latent or a set of traditional feature split latents that seem to act as a classifier for a feature like "starts with S" fail to fire in a seemingly arbitrary number of cases. What fires instead are approximately token-aligned latents with small but positive alignment with the LR probe. We say these latents are **absorbing** the feature. One case study is in Appendix A.4.

[3] `https://www.neuronpedia.org/list/cm0h1n2mt00019jdk274owq9e`

We quantify the extent to which feature absorption occurs with the metric **feature absorption rate**. We first find k feature splits for a first-letter feature using a k-sparse probe. We then find false-negative tokens that all k feature-split SAE latents fail to activate on, but which the LR probe correctly classifies, and run an integrated-gradients ablation experiment on those tokens. The ablation effect finds the most causally important SAE latents for the spelling of that token. If the SAE latent receiving the largest negative magnitude ablation effect has a cosine similarity with the LR probe above 0.025, and is at least 1.0 larger than the latent with the second highest ablation effect, we say that feature absorption has occurred. These thresholds were chosen from manual inspection of the data. We then calculate feature absorption rate as below:

$$\texttt{absorption_rate} = \frac{\texttt{num_absorptions}}{\texttt{lr_probe_true_positives}}$$

If there are more than 200 false negative per letter, we randomly sample 200 samples to estimate the number of absorptions. We see that absorption rate increases with higher sparsity and higher SAE width. Lower L0 likely pushes the SAE to absorb dense features like spelling information across multiple latents, thereby increasing feature sparsity. Feature absorption rate vs L0 for Gemma Scope SAEs layers 0-17 is shown in Figure 5b. Absorption rate by letter is shown in Appendix A.10.

Our absorption metric is not perfect, it is likely an under-estimate of the true level of feature absorption. It will not capture multiple absorbing latents activating together, or the main latents activating but very weakly. Additionally, we can't calculate absorption past layer 17 since by then attention has moved information to the final token, making ablation unavailable, as we show in Appendix A.9.

6 Discussion

Interpretability of SAE latents. We use a first-letter identification task to investigate whether SAEs extract interpretable features across different hyperparameters like size, sparsity, or layer. We find that the SAE latents we investigated were not interpretable and that varying the hyperparameters did not meaningfully change this.

One may argue it is unreasonable to judge an SAE latent against a linear probe directly optimized to perform the same classification task and that sometimes unsupervised methods can surprise us [Nanda, 2023]. We argue that interpretable latents should at minimum behave consistently with their apparent activation patterns. In our experiments, we use SAE latents that do appear to perform first letter classification, and then evaluate how well they perform this task. We validate as well that these latents causally mediated model performance on the first-letter task in Appendix A.6.

Feature absorption. We identified a form of feature splitting we call "feature absorption" related to SAE latents underperforming LR probes. It creates an interpretability illusion where a seemingly interpretable latent has arbitrary false negatives in its mainline interpretation. Lower recall poses problems for using SAEs for high-stakes classification or finding sparse circuits [Marks et al., 2024], as the number of latents needed to characterize model behavior may be much larger than expected. Occurrence of absorption even in high-recall latents suggests it's a fundamental issue.

Feature absorption is not predicted by toy-models where features do not co-occur [Bricken et al., 2023]. We suspect it is a consequence of co-occurrence between sparse and dense features. If a dense feature like "starts with letter D" always co-occurs with a more sparse feature like "dogs", the SAE can increase sparsity by absorbing the "starts with D" feature into a "dogs" latent.

It remains to be seen if we can predict or identify cases where a feature "should have activated" but does not due to absorption. Meta-SAEs, a novel method for decomposing SAE latents [Bussmann et al., 2024], may help decompose absorbed features. One interpretation is that competition exists between latents and meta-latents for activation on particular examples and reallocation of examples between SAE latents enables SAEs to interpolate between different possible decompositions.

Limitations The results presented in our paper are based on a single model (Gemma-2-2B) using one type of SAE architecture (JumpReLU). Our feature absorption metric requires having ground-truth knowledge of true labels to first train a LR probe, whereas many features of interest in a LLM lack such clear-cut ground-truth labels. Our metric uses ablation effect to ensure absorbed features causally mediate model behavior, but therefore cannot be easily used in final model layers.

7 Acknowledgements

This project was produced as part of the LASR Labs research program.

References

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes, 2017. URL https://openreview.net/forum?id=ryF7rTqgl.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, et al. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2, 2023.

Bart Bussmann, Michael Pearce, Patrick Leask, Joseph Bloom, Lee Sharkey, and Neel Nanda. Showing sae latents are not atomic using meta-saes, 2024. URL https://www.lesswrong.com/posts/TMAmHh4DdMr4nCSr5/showing-sae-latents-are-not-atomic-using-meta-saes.

Hoagy Cunningham, Logan Riggs Smith, Aidan Ewart, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=F76bwRSLeK.

Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Yoav Goldberg. Amnesic probing: Behavioral explanation with amnesic counterfactuals. *Transactions of the Association for Computational Linguistics*, 9:160–175, 2021.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, et al. Toy models of superposition. *arXiv preprint arXiv:2209.10652*, 2022.

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.

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. Dissecting recall of factual associations in auto-regressive language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12216–12235, 2023.

Wes Gurnee, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris Bertsimas. Finding neurons in a haystack: Case studies with sparse probing. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=JYs1R9IMJr.

Adam Karvonen, Benjamin Wright, Can Rager, Rico Angell, Jannik Brinkmann, Logan Riggs Smith, Claudio Mayrink Verdun, David Bau, and Samuel Marks. Measuring progress in dictionary learning for language model interpretability with board game models. In *ICML 2024 Workshop on Mechanistic Interpretability*, 2024.

Ayush Kaushal and Kyle Mahowald. What do tokens know about their characters and how do they know it? *arXiv preprint arXiv:2206.02608*, 2022.

Connor Kissane, Robert Krzyzanowski, Joseph Isaac Bloom, Arthur Conmy, and Neel Nanda. Interpreting attention layer outputs with sparse autoencoders, 2024. URL https://arxiv.org/abs/2406.17759.

Kenneth Li, Aspen K Hopkins, David Bau, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Emergent world representations: Exploring a sequence model trained on a synthetic task. *ICLR*, 2023.

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, August 2024.

Johnny Lin and Joseph Bloom. Analyzing neural networks with dictionary learning, 2023. URL https://www.neuronpedia.org. Software available from neuronpedia.org.

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. *Computing Research Repository*, arXiv:2403.19647, 2024. URL https://arxiv.org/abs/2403.19647.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, 36, 2022. arXiv:2202.05262.

Neel Nanda. Actually, othello-gpt has a linear emergent world model, mar 2023. *URL< https://neelnanda.io/mechanistic-interpretability/othello*, 2023.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. *Distill*, 5(3):e00024–001, 2020.

Chris Olah, Nicholas Turner, Adam Jermyn, and Joshua Batson. July update. https://transformer-circuits.pub/2024/july-update/index.html, 2024. URL https://transformer-circuits.pub/2024/july-update/index.html.

Bruno A Olshausen and David J Field. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision research*, 37(23):3311–3325, 1997.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=UGpGkLzwpP.

Senthooran Rajamanoharan, Tom Lieberum, Nicolas Sonnerat, Arthur Conmy, Vikrant Varma, János Kramár, and Neel Nanda. Jumping ahead: Improving reconstruction fidelity with jumprelu sparse autoencoders. *arXiv preprint arXiv:2407.14435*, 2024.

Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, pages 206–215, 2019.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017.

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, Alex Tamkin, Esin Durmus, Tristan Hume, Francesco Mosconi, 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. https://transformer-circuits.pub/2024/scaling-monosemanticity/, May 2024. Accessed on May 21, 2024.

Matthew Watkins and Joseph Bloom. Linear encoding of character-level information in gpt-j token embeddings, 2023. URL https://www.lesswrong.com/posts/GyaDCzsyQgc48j8t3/linear-encoding-of-character-level-information-in-gpt-j.

A Appendix

A.1 Glossary of Terms

Sparse Autoencoders (SAEs): Neural networks trained to reconstruct their input while enforcing sparsity in their hidden layer. In the context of this paper, SAEs are used to decompose the dense activations of language models into more interpretable features.

SAE error term: When inserting a SAE into the computation path of the model, errors in SAE reconstruction will propagate to later parts of the model and can change the model output. We refer to the error as the SAE error term, and corresponds to the difference between the SAE output and the original SAE input activation. Marks et al. [2024] introduced the idea of adding this error term back to the SAE output to ensure that the SAE does not change model output.

Latent: We refer to neurons in the hidden layer of a SAE as latents to avoid overloading the term "feature". This is in contrast to earlier work which used the term "feature" to refer to both human-interpretable concepts and SAE hidden layer neurons.

Token-aligned latent: A latent which seems to roughly fire on variants of the same token. For instance, a "Snake" token-aligned latent may fire on the tokens "Snake", "SNAKE", " snakes", etc...

L0: The mean number of non-zero latents firing in an SAE to reconstruct activations. Lower L0 means a more sparse SAE.

Feature: We use the term "feature" to refer to an idealized human-interpretable concept that the model represent in its activations and which a SAE latent may or may not represent.

Monosemantic: Referring to a feature or representation that corresponds to a single, clear semantic concept. In the context of SAEs, a monosemantic feature would ideally capture one interpretable aspect of the input.

Interpretable: A latent being interpretable is not well defined in the field, making it difficult to ensure that different authors mean the same thing when referring to SAE interpretability. When we refer to a SAE latent as being interpretable in this work, we mean that it should behave in line with how it appears to behave after inspecting its activation patterns. If an SAE latent appears to track a feature X by a reasonable inspection of its activations but has subtle deviations from this behavior in reality, we say this is not interpretable. We thus measure interpretability via classification performance when a latent appears to be a classifier over some feature.

Feature dashboard: A dashboard showing activation patterns and max-activating examples for a SAE latent. Feature dashboards are commonly used to interpret the behavior of an SAE latent.

Neuronpedia: A platform, `https://neuronpedia.org`, which hosts feature dashboards for popular SAEs [Lin and Bloom, 2023].

Feature splitting: A phenomenon in SAEs introduced by Bricken et al. [2023], where a SAE latent tracks a general feature in a narrow SAE, but splits into multiple more specific SAE latents in a wider SAE. For instance, a latent tracking "starts with L" in a narrow SAE may split into a latent tracking "starts with capital L" and a latent tracking "starts with lowercase L" in a wider SAE.

Feature absorption: A problematic form of feature splitting where a SAE latent appears to track an interpretable feature, but that latent has seemingly arbitrary exception cases where it fails to fire. Instead, an approximately token-aligned feature "absorbs" the feature direction and fires in place of the main latent.

Circuit: In the context of neural network interpretability, a circuit refers to a subgraph of neurons or features within a neural network that work together to perform a specific function or computation. The study of circuits aims to understand how different components of a neural network interact to process information and produce outputs.

Linear probe: A simple linear classifier (typically logistic regression) trained on the hidden activations of a neural network to predict some property or task. Used to assess what information is linearly decodable from the network's representations.

K-sparse probing: A variant of linear probing where only the k most important features (as determined by some selection method) are used to train the probe. This helps identify which specific neurons or features are most relevant for a given task.

Ablation study: An experimental method where a component of a system (in this case, a neuron or feature in a neural network) is removed or altered to observe its effect on the system's performance. This helps determine the causal importance of the component.

Latent 7112	Latent 7657
žda se naplaćuje naknada	LC, an aluminum boat
. E. Søli, 20	as LIFT and LF-Net. Once
a></code> </ul	latter's sister Louise, who in

Table 1: Sample max activating examples for latents 7112 and 7657 for Gemma Scope 16k, layer 0, 105 L0 from Neuronpedia. The token where the SAE feature activates is highlighted in green. Latent 7112 appears to be a lowercase "L" starting-letter latent, and latent 7657 appears to be a corresponding uppercase "L" latent.

Integrated gradients (IG): An attribution method that assigns importance scores to input features by accumulating gradients along a path from a baseline input to the actual input. In this paper, it's used as an approximation technique for ablation studies.

In-context learning (ICL): A paradigm where a language model is given examples of a task within its input prompt, allowing it to adapt to new tasks without fine-tuning. Often used with few-shot learning techniques.

Residual stream: In the context of transformer architectures, the residual stream refers to the main information flow that bypasses the self-attention and feed-forward layers through residual connections.

Logits: The raw, unnormalized outputs of a neural network's final layer, before any activation function (like softmax) is applied. In language models, logits typically represent the model's scores for each token in the vocabulary.

Activation patching: An interpretability technique where activations at specific locations in a neural network are replaced or modified to observe the effect on the network's output. This helps in understanding the causal role of different parts of the network in producing its final output.

A.2 Ablation algorithm

Algorithm 1 SAE Feature Ablation

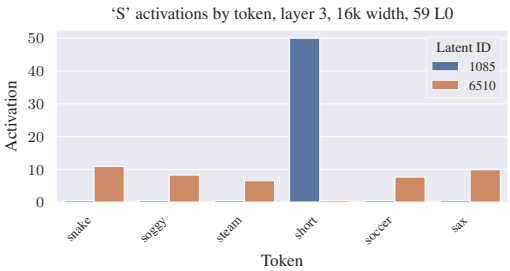
```
Insert SAE in model computation path, including error term
Define a scalar metric on the model's output distribution (e.g. difference between token logits)
Calculate baseline metric value for a test prompt with SAE and error term inserted
for each token of interest do
    for each SAE latent do
        Set the SAE latent activation to 0
        Recalculate the metric
        Compute ablation effect as (baseline metric - new metric)
        Reset the SAE latent to its original value
    end for
end for
```

A.3 Feature splitting example

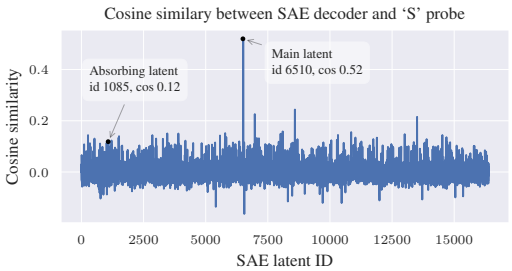
A.4 Absorption case study

The Gemma Scope layer 3, 16k width, 59 L0 SAE has a latent, 6510, which appears to act as a classifier for "starts with S", achieving an F1 of 0.81. However, this latent fails to activate on some tokens the probe can classify, and which the model can spell, such as the token `_short`.

Figure 6a shows a sample prompt containing a series of tokens that start with "S", and the activations of top SAE latents by ablation score for these tokens. The main "starts with S" latent, 6510, activates on all these tokens except `_short`. This SAE also has a token-aligned latent, 1085, which activates on variants of the word "short" (" short", "SHORT", etc...). The Neuronpedia dashboard [Lin and

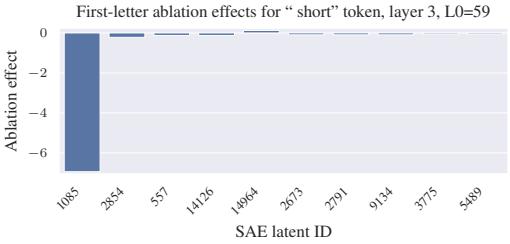


(a) Layer 3, L0=59 SAE feature activations for tokens that start with "S". The core "starts with S" feature, 6510, fails to activate on the token `_short`. The "short"-token aligned feature 1085 activates instead.

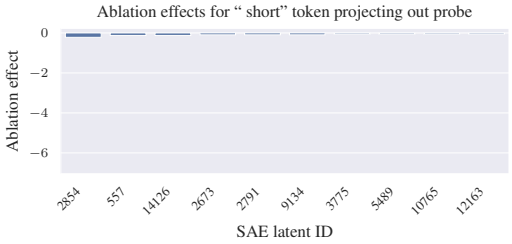


(b) Cosine similarity between layer 3, L0=59 SAE decoder and the "starts with S" probe. The main "Starts with S" latent, 6510, is clearly visible and highly probe-aligned.

Figure 6: SAE activations and cosine similarity for "starts with S" features.



(a) Ablation effect for `_short` token, indicating that feature 1085, is responsible for the "starts with S" concept for the `_short` token. The main "starts with S" latent, 6510, does not activate on the `_short` token.



(b) Ablation effect for `_short` token after removing the probe direction from latent 1085 via projection. Latent 1085 no longer appears in the plot, indicating the strong ablation effect in Figure 7a is due to its component along the probe direction.

Figure 7: Ablation effects on `_short` token before and after projecting out the probe direction

Bloom, 2023] for feature 1085 is shown in Appendix A.11. For the token `_short`, the main "starts with S" latent does not activate but the "short" latent activates instead.

Latent 1085 has a cosine similarity with the "starts with S" probe of 0.12, indicating it contains a component of the "starts with S" direction, although much smaller than the main "starts with S" latent. Cosine similarity of the SAE decoder with the "starts with S" LR probe is shown in Figure 6b. Interestingly, despite latent 1085 having only about $1/5$ the cosine similarity with the probe as the main latent 6510, we see it activates with about 5 times the magnitude of latent 6510 on the `_short` token, thus contributing a similar amount of the "starts with S" probe direction to the residual stream.

We conduct an ablation experiment on the `_short` token, shown in Figure 7a, and see that latent 1085 has a dramatically larger ablation effect compared with all other SAE features. This suggests latent 1085 is causally responsible for the model knowing that `_short` starts with S.

Is it possible that the probe projection is not the causally important component of feature 1085? We conduct another ablation experiment, except now we remove the probe direction from feature 1085 via projection before ablation. The results of this ablation experiment are shown in Figure 7b. After removing the probe component from feature 1085, it no longer has a significant ablation effect. Thus we know the probe projection of feature 1085 is responsible for model behavior.

These experiments show the "starts with S" feature has been "absorbed" by the token-aligned latent 1085, likely along with other semantic concepts related to the word "short". After observing that the main "starts with S" latent 6510 activates on most tokens that begin with "S", it may be tempting to conclude this latent tracks the interpretable feature of beginning with the letter "S". However, this latent quietly fails to activate on the `_short` token, leading us to a false sense of understanding.

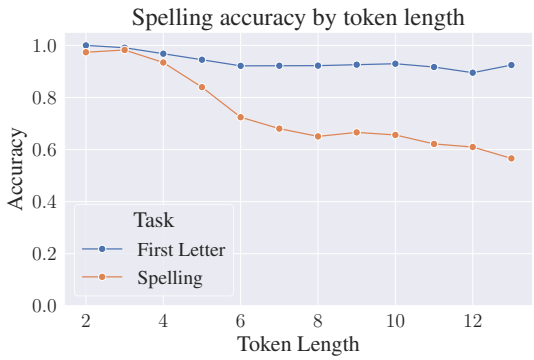


Figure 8: Baseline performance for Gemma-2-2B on first-letter identification and full-token spelling by token length.

A.5 How good is Gemma-2 on character identification tasks?

We evaluate how well can Gemma-2-2B identify the first letter or all the letters in a token (spelling the full token). We evaluate the accuracy of the model on all tokens in the LR probe validation set with a prompt containing 10 in-context examples selected at random from the full vocabulary. Our results are shown in Figure 8.

We see that performance on the first-letter identification task is high throughout token length, while the full-word spelling performance decreases as the length of the token increases.

A.6 Intervening on the first letter

If the model is using the identified SAE latents for predicting the first letter we should also be able to change what first letter it predicts just by changing the activations. For this experiment we use the SAE latents most cosine similar with the LR probe for the true first letter and for a new randomly selected letter. We take the intermediate activations of Gemma-2-2B in the residual stream and encode them using the SAE. Then we zero out the activation of the SAE latent associated with the original letter and change the activation of the SAE latent associated with the new letter into the average activation it has on tokens starting with this new letter.

Editing works better with latents from the narrower 16k SAE compared to the 65k, with the best L0s in the 75-150 range. This corresponds to the observed pattern of these SAE latents having higher F1 scores for classification. We report the results in Figure 9. The best SAEs on the layers 7-9 can achieve a substantial replacement, but note that the averages hide variance across individual tokens, where some get edited completely and others get unaffected. The edit success also varies based on the true first letter and the random new letter; for illustration we show a breakdown by letter for two specific SAEs in layer 7 in Figure 10.

A.7 Probe cosine similarity vs k=1 sparse probing

The first step when searching for a SAE feature that acts as a first-letter classifier involves searching for SAE feature which best acts as a classifier. In Figure 2, we achieve this by first training a LR probe on the first-letter task and using cosine similarity between that probe and the SAE encoder to find the best feature for the first-letter task. We also investigated using k-sparse probing with k=1 to select the best SAE feature instead. This involves training a linear probe with L1 loss and selecting the feature with the highest positive weight from the probe.

We find that both k=1 sparse probing yield nearly identical results, as seen in Figures 11 and 12. Additionally Figure 13 shows the cosine similarity of the LR probe with each SAE feature by letter for the canonical Gemma Scope layer 0 16k width SAE. In most cases there is an obvious probe-aligned feature. Likely any reasonable method of feature selection will find the same feature for these cases. We thus decided to use cosine similarity between the SAE encoder and a LR probe as our selection criteria for single SAE features as this is a simpler metric and less computationally intensive to compute.

(a) Comparing success in editing out the true first letter and making the model predict a randomly selected new letter across layers 0-9 for all 16k and 65k Gemma Scope SAEs.

(b) Comparing the edit success with the top SAE feature across all L0s for 16k and 65k widths across layers 0-9. The best performance seems to be occurring for L0 between 75 and 150.

Figure 9: Comparison of Edit success by Layer and L0

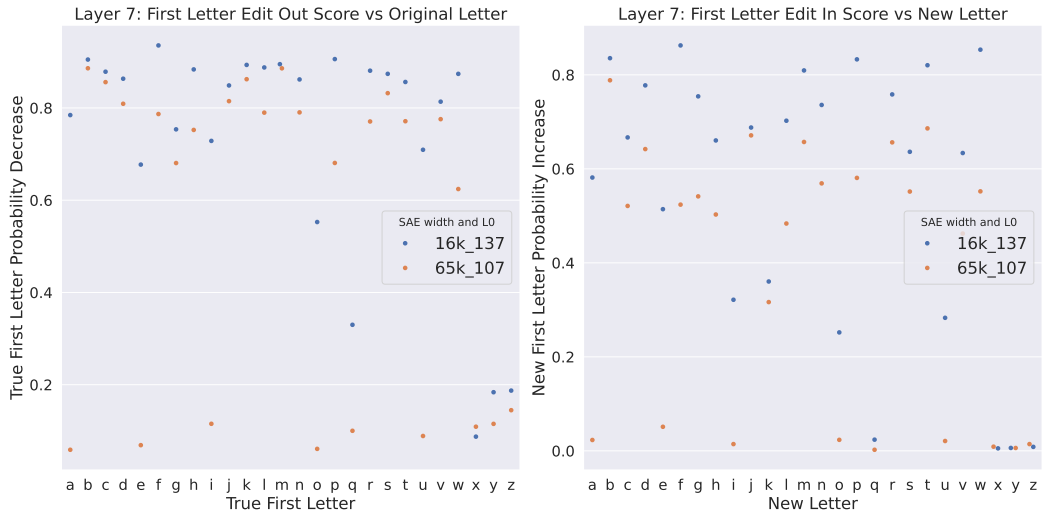
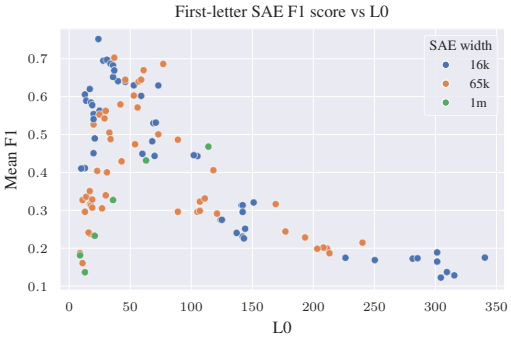


Figure 10: Comparing the edit success broken down by the letter at layer 7 for two SAEs; SAE width 16,000 and L0 of 137 and SAE width 65,000 and L0 of 107. For each original letter we draw a sample of 100 tokens and average the decrease in probability of the correct first letter and increase in probability of a new random letter.

A.8 Precision, recall, and F1 score for the first-letter task

We evaluated precision, recall, and F1 score for the first-letter classification task, and found that the precision and recall vary depending on the L0 of the SAE. Low L0 SAEs learn high precision, low recall features, while high L0 SAEs learn low precision, high recall features. These results are shown in Figure 14. We thus chose to use F1 score as our core metric in this paper to balance precision and recall as many if the SAEs we tested have extreme values in either precision or recall.

While it may appear that there is an optimal L0 from looking at aggregate statistics across letter, we find that breaking down the F1 vs L0 plot by letter reveals that the optimal L0 appears different for different letters, with low frequency letters like z actually having the best F1 score at the lowest L0, while other letters instead have an optimal L0 around 30-50. Figure 15 shows these results broken down by letter.

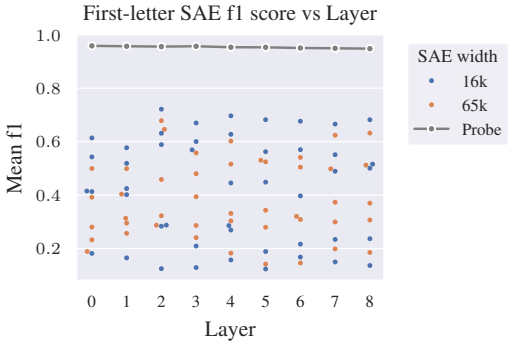


(a) Mean F1 score on first-letter classification task using top SAE encoder feature by cosine similarity with the LR probe vs L0 for all Gemma Scope SAEs layers 0-9.

(b) Mean F1 score on first-letter classification task using k=1 sparse probing to select the SAE feature for the first-letter classification task vs L0 for all Gemma Scope SAEs layers 0-9.

Figure 11: Comparison of LR probe cosine similarity and k=1 sparse probing vs l0

(a) Mean F1 score on first-letter classification task using top SAE encoder feature by cosine similarity with the LR probe vs layer for all Gemma Scope SAEs layers 0-9.



(b) Mean F1 score on first-letter classification task using k=1 sparse probing to select the SAE feature for the first-letter classification task vs layer for all Gemma Scope SAEs layers 0-9.

Figure 12: Comparison of LR probe cosine similarity and k=1 sparse probing vs layer

A.9 Causal interventions and absorption

In this work, we rely on causal interventions like ablation experiments to verify that SAE latents have a causal impact on model behavior. In these experiments for spelling tasks, we set up an ICL prompt to elicit spelling information from the model, for instance the ICL prompt below:

```
tartan has the first letter: T
mirth has the first letter: M
dog has the first letter:
```

In this ICL prompt, we would apply an SAE and train LR probes on the `_dog` token position, and expect that the model will output the token `_D`. When we intervene on the `_dog`, we can track the causal changes to model outputs by applying a metric to the output logits, e.g. checking how our intervention increases or decreases the `_D` logit relative to other letters.

We use these interventions as part of our absorption metric to ensure that when we claim that "absorption" is occurring, we verify that the absorbing feature has a causal impact on model outputs. This is stronger evidence than only noting a cosine similarity between the absorbing feature, but this means that our absorption metric cannot classify absorption at later model layers.

During a LLM forward pass, the model first collects relevant information on a token in that token position, and attention heads then move relevant information from earlier tokens to later tokens [Geva et al., 2023, Meng et al., 2022]. If we assess ablation effect at layers after which model attention has already pulled relevant information from the subject tokens into the final output token, the ablation

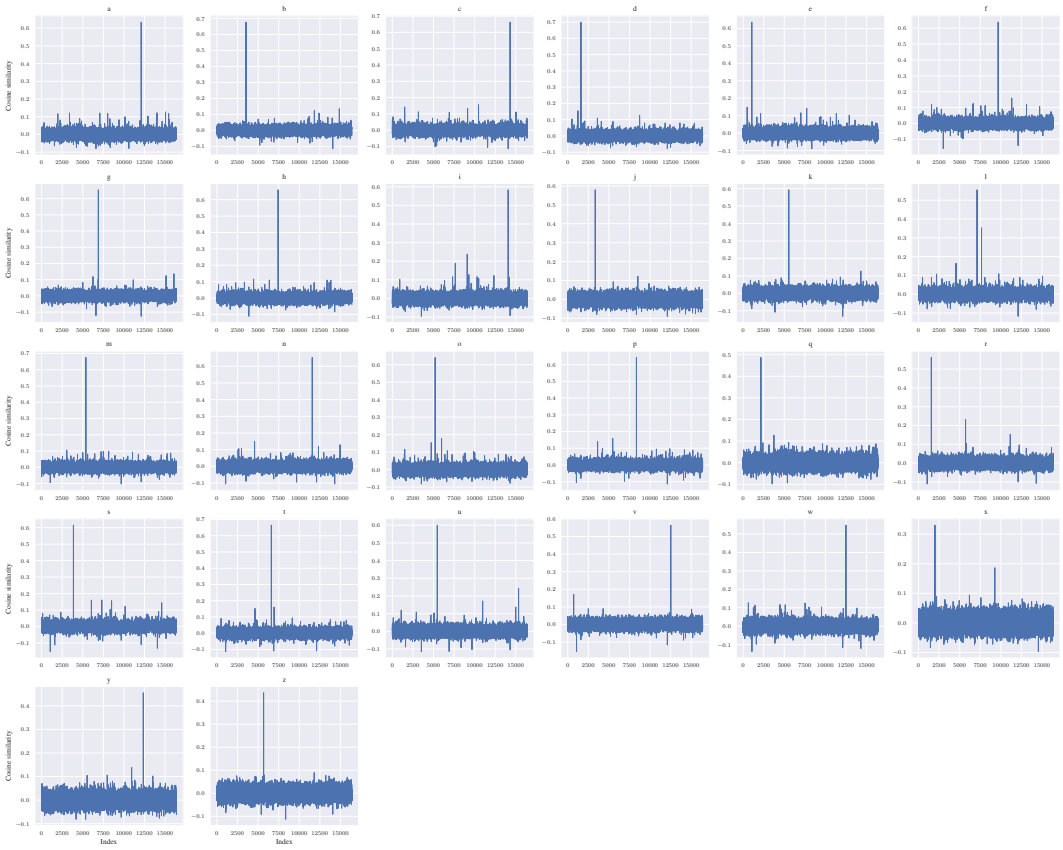
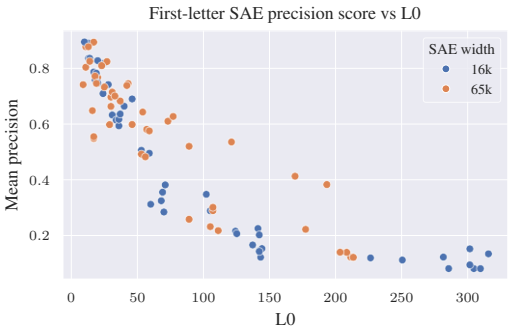
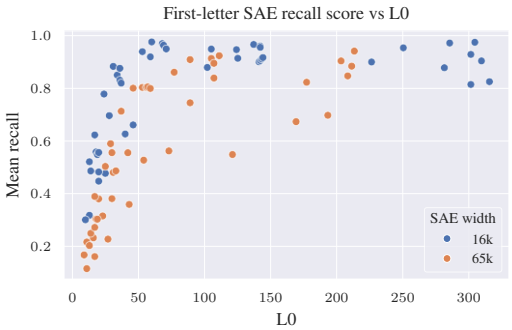


Figure 13: Decoder cosine similarities with the LR probe by letter, Gemma Scope 16k layer 0 l0=105. Most letters have one or two obvious SAE features which align with the probe.



(a) Mean precision on first-letter classification task vs L0 for all Gemma Scope SAEs layers 0-9. Features are selected via k=1 sparse probing



(b) Mean recall on first-letter classification task vs L0 for all Gemma Scope SAEs layers 0-9. Features are selected via k=1 sparse probing

Figure 14: Comparison of precision and recall vs l0

effect will be 0. For Gemma 2 2B on the first-letter spelling task, we find this movement of first-letter spelling information occurs around layer 18.

Figure 16 shows an activation patching experiment [Meng et al., 2022] on a sample first-letter spelling prompt. In this experiment, we see that near layer 18 the model moves first-letter spelling information from the subject token to the prediction token.

As a result, our feature absorption metric will not function past layer 18 in Gemma 2 2B, and we thus focus on layers 0-17 for our analysis of feature absorption. We believe that feature absorption is still occurring in SAEs past layer 18, but we lose the ability to make causal claims that the absorbing

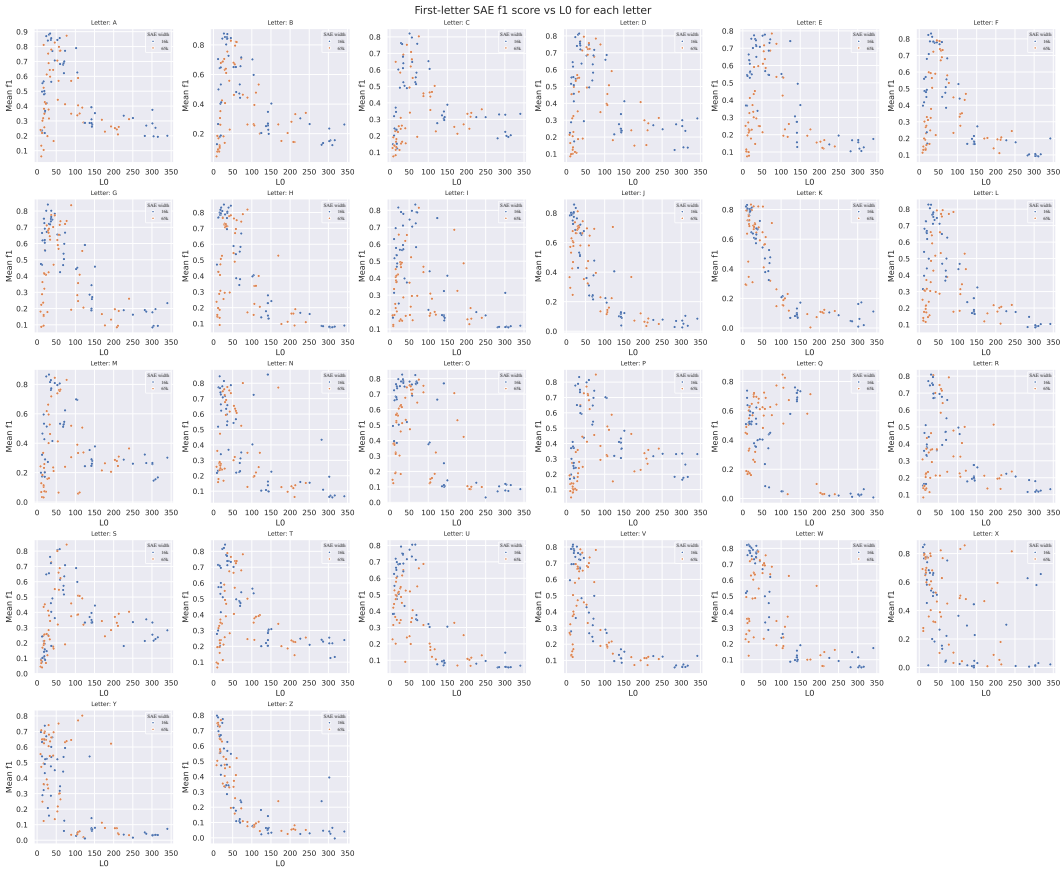


Figure 15: F1 vs L0 by letter. SAE features are picked using k=1 sparse probing.

features are used by the model to make predictions. Given that this paper is trying to highlight the existence of feature absorption, we felt it is more important to have a metric which is robust and has the backing of causal analysis but which cannot be used at all model layers. Future work may make a different trade-off and choose a feature absorption metric which can work at all model layers, for instance relying only on cosine similarity between absorbing features and a LR probe to determine absorption.

A.10 Additional plots

In this section, we include additional plots that are too large to fit in the main body of the paper.

A.11 Feature dashboards

We include feature dashboard screenshots from Neuronpedia for some prominent latents mentioned in this work. Figure 18 shows a dashboard for Gemmascope layer 3, latent 1085, which is a token-aligned latent firing on variations of the word `_short` and we find absorbs the "starts with S" direction. Figure 19 shows latent 6510 from the same layer which should be the main "starts with S" latent.

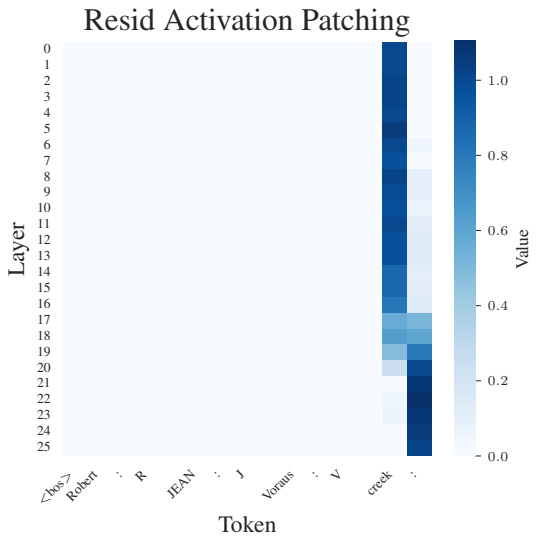


Figure 16: Residual stream attribution patching for a sample first-letter spelling prompt, Gemma 2 2B. After around layer 18, model attention moves the relevant spelling information from the source token to the prediction location.

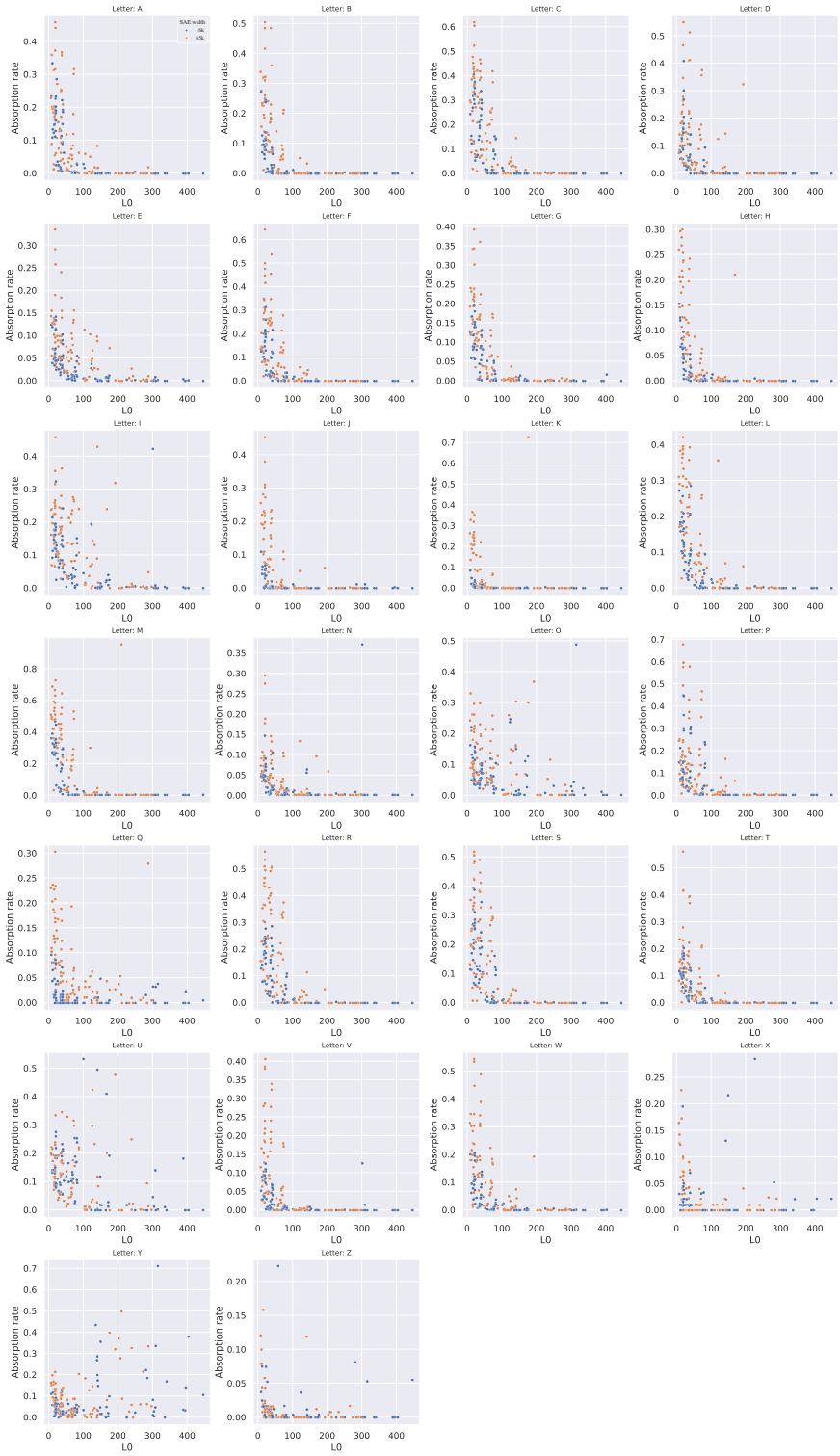


Figure 17: Absorption rate vs L0 by letter, layers 0-17. We see a wide variance in which letters are absorbed by which SAEs.

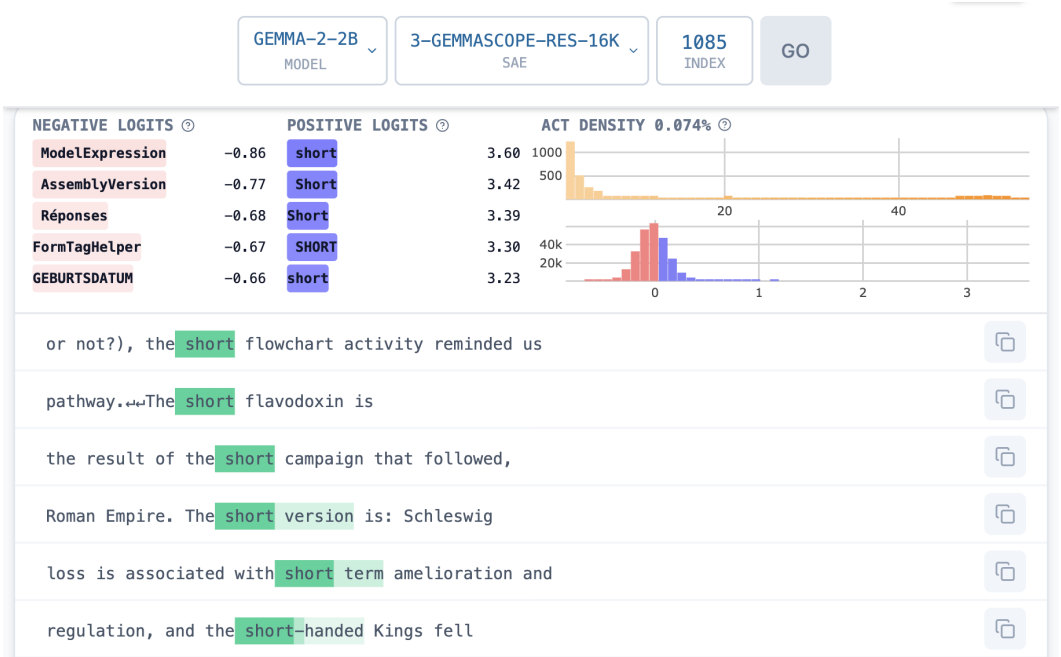


Figure 18: Neuronpedia dashboard for Gemma Scope layer 3, latent 1085. This latent is a token-aligned latent for `_short` tokens. This latent absorbs the "starts with S" direction.

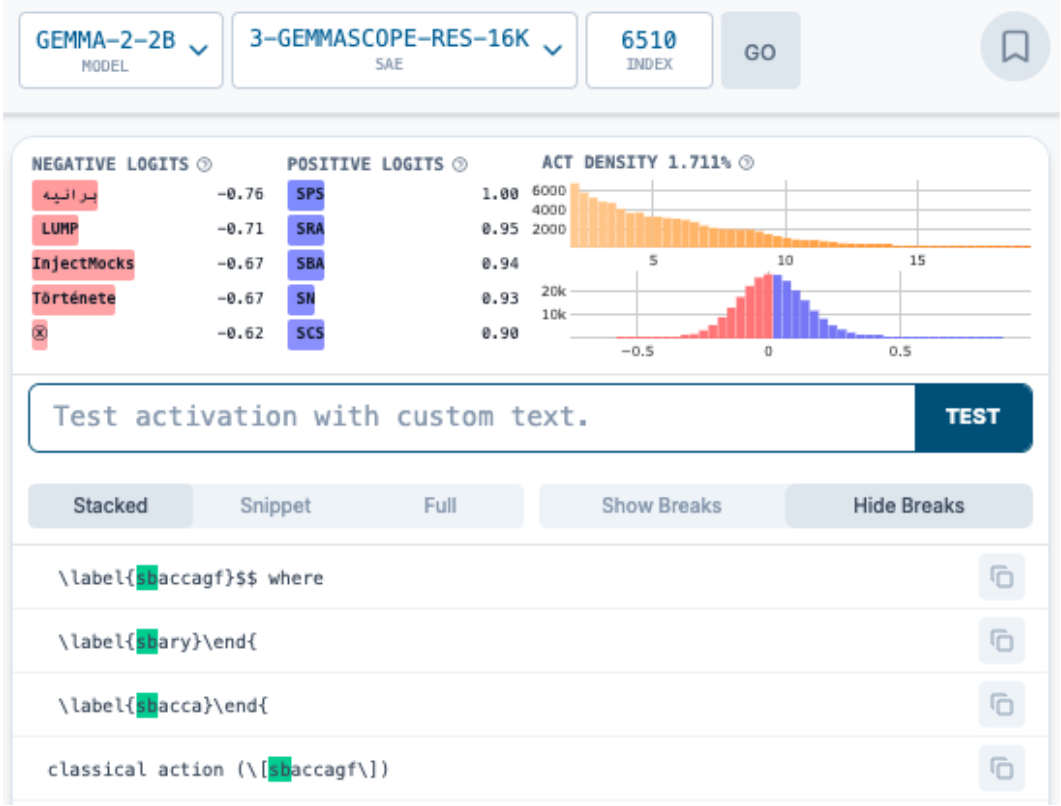


Figure 19: Neuronpedia dashboard for Gemma Scope layer 3, latent 6510. This latent should be the main "starts with S" latent.