has learned a perfect representation of feature 0, but the encoder has a hole in its recall. Latent 0 fires if feature 0 is active but not feature 1. This is exactly the sort of gerrymandered feature firing pattern we will see later in real SAEs in Section 5.2 - the encoder has learned to stop the latent firing on specific cases where it looks like it should be firing. In addition, we see that latent 1, which tracks feature 1, has absorbed the feature 0 direction. This results in latent 1 representing a combination of feature 0 and feature 1. We see that the independently firing features 2 and 3 are untouched - the SAE still learns perfect representations of these features. These results are summarized in Table 3. We explore absorption in more toy settings in Appendix A.3.

Proof: hierarchical features cause absorption We further provide an analytical proof that in the hierarchical setup described above, feature absorption decreases SAE loss in Appendix A.2.

4 Experimental setup

Our experiments on LLM SAEs focus 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 include the SAE error term [21] to avoid changing model output.

To determine the causal effect of SAE latents on the first-letter identification task we conduct ablation studies. We use a metric consisting 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 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:

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

We discuss this metric and alternative formulations further in Appendix A.10.

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.

We use the base Gemma-2-2B model for most of our studies, along with the full set of Gemma Scope residual stream SAEs of width 16k and 65k released by Deepmind [19]. We also evaluate absorption on our own SAEs trained on Qwen2 0.5B [32] and Llama 3.2 1B [6].

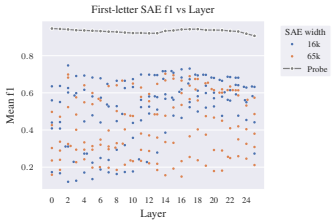
5 Results

Our results are divided into four sections. First, we compare the performance of linear probes with SAE latents on recovering first-character information from model activations, showing that despite appearing to track first letter features, a wide variety of precision / recall is achieved. Second, we motivate our definition of feature absorption with a case-study, emphasizing how an absorbing latent can unexpectedly causally mediate first letter information whilst the first-letter latent (unexpectedly) fails to fire. Next, we attempt to quantify feature splitting and feature absorption, showing that tuning of hyper-parameters may partially assist but not fully alleviate feature absorption.

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 observed that for each probe, there was clearly one or at most a couple of outlier SAE latents with high probe cosine similarity. Full plots of cosine similarity vs letter are shown in Appendix A.8.

We also tried using k=1 sparse probing [12] to identify SAE latents, and found this gives similar results. Further comparison of using k=1 sparse probing vs encoder cosine similarity to identify latents is explored in Appendix A.7.



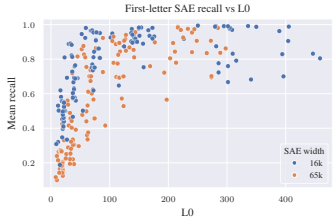
(a) Mean F1 score using top SAE encoder latent by layer for Gemma Scope SAEs and LR probe.

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

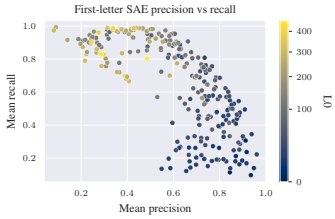
(c) Mean F1 score vs L0 for Gemma Scope SAEs layers 13-25. SAEs learn cleanest "first-letter" features with L0 near 50-100.

Figure 4: Comparison of F1 scores for first-letter classification tasks

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



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



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

Figure 5: Precision and recall vs L0 for first-letter classification tasks

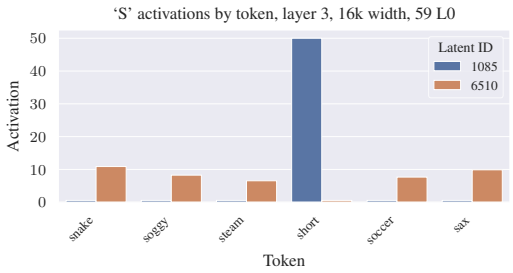
We observe wide variance in the performance of Gemma Scope SAEs at the first-letter identification task, but no SAE matches LR probe performance. We show the mean F1 score by layer as well as the F1 score of the LR probe in Figure 4a. We further investigate the F1 score of these SAE encoder latents as a function of L0 and SAE width in Figures 4b and 4c.

Whether or not an SAE learns a clear "first-letter" latent for each letter is highly dependent on L0, with low L0 SAEs tending to learn high-precision low-recall latents, and high L0 SAEs learning low-precision high-recall latents (Figure 5). 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.

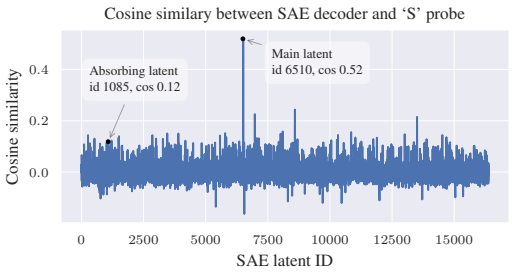
5.2 Why do SAE latents underperform?

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 [20] for latent 1085 is shown in Appendix A.15. For the token `_short`, the main "starts with S" latent does not activate but the "short" latent activates instead.

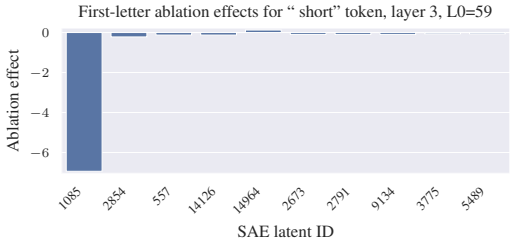


(a) Layer 3, L0=59 SAE latent activations for tokens that start with "S". The core "starts with S" latent, 6510, fails to activate on the token `_short`. The "short"-token aligned latent 1085 activates instead.

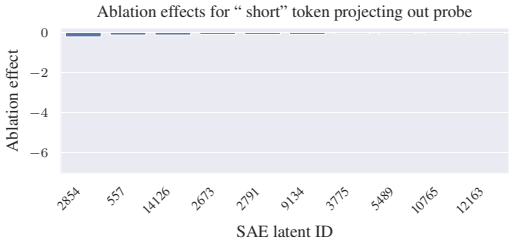


(b) Cosine similarity between layer 3, L0=59 SAE decoder and the "starts with S" probe. The main "Starts with S" latent, 6510, is clearly visible and highly probe-aligned.

Figure 6: Comparison of SAE latent activations and cosine similarity for tokens starting with "S"



(a) Ablation effect for `_short` token, indicating that latent 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

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 study the ablation effect of each SAE latent on the `_short` token, shown in Figure 7a, and see that latent 1085 has a dramatically larger ablation effect compared with all other SAE latents. 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 latent 1085? We conduct another ablation effect experiment, except now we remove the probe direction from latent 1085 via projection before ablation. The results of this experiment are shown in Figure 7b. After removing the probe component from latent 1085, it no longer has a significant ablation effect. Thus we know the probe projection of latent 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.

Here we clearly see feature absorption. The seemingly interpretable SAE latent 6510 fails to activate on arbitrary positive examples, and instead the feature is "absorbed" into more specific latents.

Feature absorption is likely a logical consequence of SAE sparsity loss. If a dense and sparse feature co-occur, absorbing the dense feature into a latent tracking the sparse feature will increase sparsity.