AUTOMATICALLY INTERPRETING MILLIONS OF FEATURES IN LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

While the activations of neurons in deep neural networks usually do not have a simple human-understandable interpretation, sparse autoencoders (SAEs) can be used to transform these activations into a higher-dimensional latent space which may be more easily interpretable. However, these SAEs can have millions of distinct latent features, making it infeasible for humans to manually interpret each one. In this work, we build an open-source automated pipeline to generate and evaluate natural language explanations for SAE features using LLMs. We test our framework on SAEs of varying sizes, activation functions, and losses, trained on two different open-weight LLMs. We introduce five new techniques to score the quality of explanations that are cheaper to run than the previous state of the art. One of these techniques, intervention scoring, evaluates the interpretability of the effects of intervening on a feature, which we find explains features that are not recalled by existing methods. We propose guidelines for generating better explanations that remain valid for a broader set of activating contexts, and discuss pitfalls with existing scoring techniques. We use our explanations to measure the semantic similarity of independently trained SAEs, and find that SAEs trained on nearby layers of the residual stream are highly similar. Our large-scale analysis confirms that SAE latents are indeed much more interpretable than neurons, even when neurons are sparsified using top-k postprocessing. We hope our open-source framework will improve future evaluations of the interpretability of SAEs and enable more work on this front.

1 INTRODUCTION

Large language models (LLMs) have reached human level performance in a broad range of domains (OpenAI, 2023), and can even be leveraged to develop agents (Wang et al., 2023) that can strategize (Bakhtin et al., 2022), cooperate and develop new ideas (Lu et al., 2024; Shaham et al., 2024). At the same time, we understand little about the internal representations driving their behavior. Early mechanistic interpretability research focused on analyzing the activation patterns of individual neurons (Olah et al., 2020; Gurnee et al., 2023; 2024). Due to the large number of neurons to be interpreted, automated approaches were proposed (Bills et al., 2023), where a second LLM is used to propose an explanation given a set of neuron activations and the text snippets it activates on, in a process similar to that of generating a human label. But research has found that most neurons are "polysemantic," activating in contexts that can be very different(Arora et al., 2018; Elhage et al., 2022). The Linear Representation Hypothesis (Park et al., 2023) posits that human-interpretable concepts are encoded in *linear combinations* of neurons. A significant branch of current interpretability work focuses on extracting these features and disentangling them (Bereska & Gavves, 2024).

Sparse autoencoders (SAEs) were proposed as a way to address polysemanticity (Cunningham et al., 2023). SAEs consist of two parts: an encoder that transforms activation vectors into a sparse, higher-dimensional latent space, and a decoder that projects the latents back into the original space. Both parts are trained jointly to minimize reconstruction error. SAE latents were found to be interpretable and potentially more monosemantic than neurons (Bricken et al., 2023; Cunningham et al., 2023). Recently, a significant effort was made to scale SAE training to larger models, like GPT-4 (Gao et al., 2024) and Claude (Templeton et al., 2024), and they have become an important interpretability tool for LLMs.

Training an SAE yields a large number of sparse latent features, each of which needs a natural language explanation. In this work, we introduce an automated framework that uses LLMs to generate an explanation for each latent in an SAE. We use this framework to explain millions of latents across multiple models, layers, and SAE architectures. We also propose new ways to evaluate the quality of explanations, and discuss the problems with existing approaches.

Although other repositories of explanations for SAE features already exist (Lin & Bloom, 2023), we believe it would be helpful for the community to provide high quality explanations, using open source models, in a way that is reproducible. This kind of exhaustive explanatory work could enable more downstream SAE applications utilizing SAE capabilities like steering and concept localization.

1.1 Explaining SAE latents in LLMs

Explaining the latents of SAEs trained on models like Llama 3.1 7b or Gemma 2 9b requires the generation of millions of explanations. As an example, the most extensive open-source set of SAEs available, Gemmascope (Lieberum et al., 2024), includes SAEs for all layers of Gemma 2 9b and Gemma 2 2b and would require explaining tens of millions of latents. Doing the same for bigger models like Llama 3.1 70b or 405b could easily reach hundreds of millions, if not billions, of latents to be explained, meaning that optimizing the process to obtain these explanations is essential.

2 Related work

One of the first approaches to automated interpretability focused on explaining neurons of GPT-2 using GPT-4 (Bills et al., 2023). GPT-4 was shown examples of contexts where a given neuron was active and was tasked to provide a short explanation that could capture the activation patterns. To evaluate if a given explanation captured the behavior of the neuron, GPT-4 was tasked to predict the activations of the neuron in a given context having access to that explanation. The explanation is then scored by how much the simulated activations correlate with the true activations.

A similar approach was used in Templeton et al. (2024) to explain SAE latents of Claude. In general, current approaches focus on collecting contexts together with latent activations from the model to be explained, and use a larger model to find patterns in activating contexts.

Following those works, other methods of evaluating explanations have been proposed, including asking a model to generate activating examples and measuring how much the neuron activates (Kopf et al., 2024; Hernandez et al., 2022). More recently, "interpretability agents" have been proposed, iteratively doing experiments to find the best explanations of vision neurons (Shaham et al., 2024).

On the other end of the spectrum, a potentially cheaper version of automated interpretability has been proposed, where the model that is being explained doubles as an explanation generation model (Kharlapenko et al., 2024). A prompt querying the meaning of a single placeholder token is passed to the model and latent activations are patched into its residual stream at the position of the placeholder token during execution, generating continuations related to the latent. This technique is inspired by earlier work on Patchscopes (Ghandeharioun et al., 2024) and SelfIE (Chen et al., 2024).

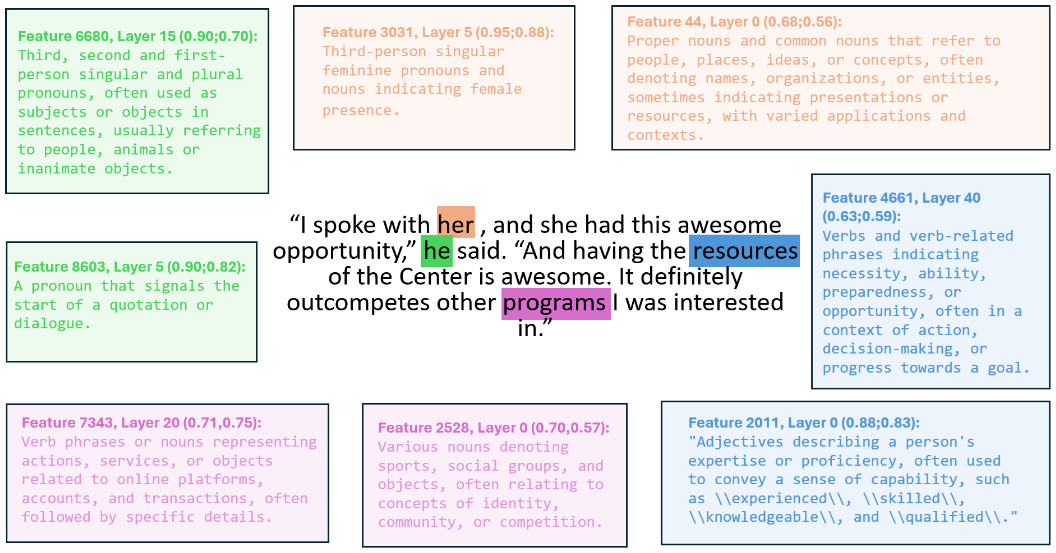
3 Methods

3.1 Collecting activations

We collected latent activations from the SAEs over a 10M token sample of RedPajama-v2 (RPJv2; Computer 2023). This is the same dataset that we used to train our Llama 3.1 8b SAEs, and consists of a data mix similar to the Llama 1 pretraining corpus.

We collected batches of 256 tokens starting with the beginning of sentence token (BOS).[1] The contexts used for activation collection are smaller than the contexts used to train the SAEs, and we find that on average, 30% of the latents of the 131k latent, per layer, Gemma 2 9b SAE don't activate more than 200 times over these 10M tokens and 15% don't activate at all. When we consider the

[1]We throw out the activations on the BOS token when generating explanations for Gemma, as we were told in personal communication that these SAEs were not trained on BOS activations.


Feature 6680, Layer 15 (0.90;0.70): Third, second and first-person singular and plural pronouns, often used as subjects or objects in sentences, usually referring to people, animals or inanimate objects.

Feature 3031, Layer 5 (0.95;0.88): Third-person singular feminine pronouns and nouns indicating female presence.

Feature 44, Layer 0 (0.68;0.56): Proper nouns and common nouns that refer to people, places, ideas, or concepts, often denoting names, organizations, or entities, sometimes indicating presentations or resources, with varied applications and contexts.

"I spoke with her , and she had this awesome opportunity," he said. "And having the resources of the Center is awesome. It definitely outcompetes other programs I was interested in."

Feature 8603, Layer 5 (0.90;0.82): A pronoun that signals the start of a quotation or dialogue.

Feature 4661, Layer 40 (0.63;0.59): Verbs and verb-related phrases indicating necessity, ability, preparedness, or opportunity, often in a context of action, decision-making, or progress towards a goal.

Feature 7343, Layer 20 (0.71,0.75): Verb phrases or nouns representing actions, services, or objects related to online platforms, accounts, and transactions, often followed by specific details.

Feature 2528, Layer 0 (0.70,0.57): Various nouns denoting sports, social groups, and objects, often relating to concepts of identity, community, or competition.

Feature 2011, Layer 0 (0.88;0.83): "Adjectives describing a person's expertise or proficiency, often used to convey a sense of capability, such as \\experienced\\, \\skilled\\, \\knowledgeable\\, and \\qualified\\."

Figure 1: **SAE latents explanations in a random sentence**. To visualize the latent explanations produced, we select a sentence taken from the RPJv2 dataset. We selected 4 tokens in different positions in that sentence and filter for latents that are active in different layers. Then we randomly select active latents and their corresponding explanations to display. We display the detection and fuzzing scores of each explanation, which indicate how well it explains other examples in the dataset (see section 3 for details on these scores). The features selected had high activation, but were not cherry-picked based on explanations or scores.

training context length of 1024, only 5% of latents don't fire. When using a closer proxy of the training data, the "un-copyrighted" Pile, we find that the number of latents that activate fewer than 200 times decreases to 15%, and that only 1% don't activate at all, even when considering contexts of size 256. Interestingly, in the case of the 16k latent Gemma 2 9b SAE only around 10% of latents activate fewer than 200 times on the RPJv2 sample, suggesting that the larger 131k latent SAE learned more dataset specific latents than its smaller cousin.

3.2 Prompting the explaner

Our approach follows Bills et al. (2023) in showing the explainer model examples sampled from different quantiles, but uses a more "natural" prompt where the activating example is shown whole, with the activating tokens emphasized and the strength of the activation shown after the example. See Appendix A.1 for the full prompt.

The activating example is selected to only have 32 tokens, irrespective of the position of the activating tokens in the prompt. We found that using COT, at least when explaining using Llama 70b, does not significantly increase the quality of the explanations, while significantly increasing the compute and time required to generate explanations. For this reason, we have not used it for our main experiments. Showing such short contexts to the explainer model hinders the correct identification of latents with complex activation patterns.

3.3 Maximally activating contexts

We find that randomly sampling from a broader set of examples leads to explanations that cover a larger set of activating examples, sometimes to the detriment of the top activating examples, see fig. 3. Sampling from top examples often generates more concise and specific explanations that accurately describe the top activating examples but fail to capture the whole distribution. On the other hand, uniformly sampling examples that activate the feature, or ensuring that chosen examples cover all activation strength quantiles, can lead to explanations that are too broad and that fail to capture a meaningful explanation. For examples of these types of failure modes, see fig. A1 and the discussion in the appendix A.2.

Focusing on maximally activating contexts to generate and score explanations biases the types of explanations found and the types of explanations that are highly scored.

3.4 Automatically interpreting interventions

Automatic interpretability methods, including most of those explored in this paper, typically look for correlations between activation of a feature and some natural-language property of the input. However, some features are more closely related to what the model will output. For example, we find a feature[2] whose activation causes the model to output words associated with reputation but does not have a simple explanation in terms of inputs.

We define an *output feature* as a feature that causes a property of the model's output that can be easily explained in natural language. See sec 3.5.5 for the definition we use in scoring.

Output features can also be described in terms of their correlation with inputs. For example, the "reputation" feature activates in contexts where likely next tokens relate to reputation. However, explaining output features in terms of causal influence on output has two advantages.

1. **Scalability**. Output features are easier to describe in terms of effect on output because the explainer only needs to notice a simple pattern of output. The pattern of inputs this feature correlates with is more complex. Explaining a feature by correlating it with inputs requires approximating the computation of the subject model leading up to that feature, which may be challenging when subject models are highly capable and performing difficult tasks. Some features' influence on output, however, might remain easily explainable.
2. **Causal evidence**. We might like to know that a feature causes a property of the model's output so that we can steer the model. Previous work has shown that existing auto-interpretability scores fail to accurately capture how well a given explanation can predict the effect of intervening on a given neuron (Huang et al., 2023). Further, prior work has argued that causal evidence is more robust to distribution shifts (Bühlmann, 2018; Schölkopf et al., 2012).

3.5 Scoring methods

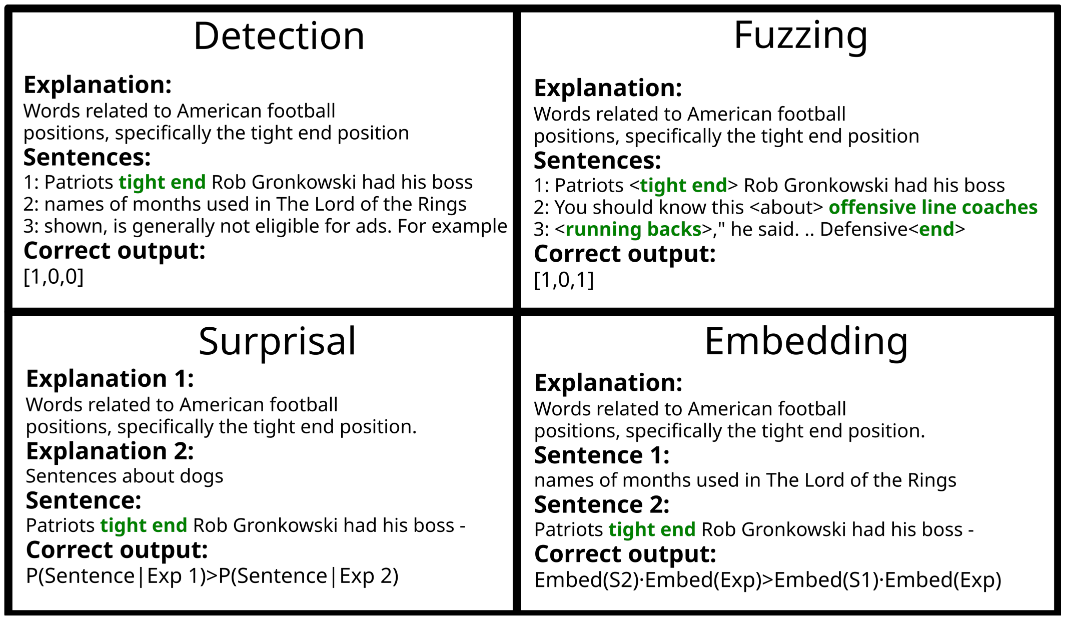
Being able to efficiently evaluate explanations of SAE latents is important for a few reasons. First, practitioners would like to know how faithful each explanation is to the actual behavior of the network. Some latents may not be amenable to a simple explanation, and in these cases we expect the auto-interpretability pipeline to output an explanation with poor evaluation metrics. Secondly, we can use evaluations as a feedback signal for the training of explainer models and tuning hyperparameters in the pipeline. Finally, some SAEs may simply be poor quality overall, and the aggregate evaluation metrics of the SAE's explanations can be used to detect this.

The quality of SAE explanations has so far been measured via simulation scoring, a technique introduced by Bills et al. (2023) for evaluating explanations of neurons. It involves asking an explainer model to "simulate" a feature in a set of contexts, which means predicting how strongly the feature should activate at each token in each context given an explanation. The Pearson correlation between the simulated and true activations is the simulation score. The standard in the literature is to sample contexts from a "top-and-random" distribution that mixes maximally activating contexts and contexts sampled uniformly at random from the corpus. While oversampling the top activating contexts introduces bias, it is used as a cheap variance reduction technique, given that simulation scoring over hundreds of examples per feature would be expensive (Cunningham et al., 2023).

In this work, we take a slightly different view of what makes for a good explanation: an explanation should serve as a *binary classifier* distinguishing activating from non-activating contexts. The reasoning behind this is simple. Given the highly sparse nature of SAE latents, most of their variance could be captured by a binary predictor that predicts the mean nonzero activation when the latent is

[2]Feature 157 of Gemma 2 9b's layer 32 autoencoder with 131k latents and an average L0 norm of 51. It has a detection score of 0.6.

expected to be active, and zero otherwise. In statistics, zero-inflated phenomena are often modeled as a mixture distribution with two components: a Dirac delta on zero, and another distribution (e.g. Poisson) for nonzero values.



Detection

Explanation:
Words related to American football positions, specifically the tight end position
Sentences:
1: Patriots **tight end** Rob Gronkowski had his boss
2: names of months used in The Lord of the Rings
3: shown, is generally not eligible for ads. For example
Correct output:
[1,0,0]

Fuzzing

Explanation:
Words related to American football positions, specifically the tight end position
Sentences:
1: Patriots <**tight end**> Rob Gronkowski had his boss
2: You should know this <about> **offensive line coaches**
3: <**running backs**>," he said. .. Defensive<**end**>
Correct output:
[1,0,1]

Surprisal

Explanation 1:
Words related to American football positions, specifically the tight end position.
Explanation 2:
Sentences about dogs
Sentence:
Patriots **tight end** Rob Gronkowski had his boss -
Correct output:
P(Sentence|Exp 1)>P(Sentence|Exp 2)

Embedding

Explanation:
Words related to American football positions, specifically the tight end position.
Sentence 1:
names of months used in The Lord of the Rings
Sentence 2:
Patriots **tight end** Rob Gronkowski had his boss -
Correct output:
Embed(S2)·Embed(Exp)>Embed(S1)·Embed(Exp)

Figure 2: **The new proposed scoring methods.** In **detection** scoring, the scorer model is tasked with selecting the set of sentences that activate a given latent given an explanation. In this work, we show 5 examples at the same time, and each has an identical probability of being a sentence that activates the latent, independent of whether any other example also activates the latent. The activating tokens are colored in green for display, but that information is not shown to the scorer model. For **fuzzing** scoring, the scorer model is tasked with selecting the sentences where the highlighted tokens are the tokens that activate a target latent given an explanation of that latent. In **surprisal** scoring, activating and non-activating examples are run through the model and the loss over those sentences is computed. Correct explanations should decrease the loss in activating sentences compared to a generic explanation, but shouldn't significantly decrease the loss in non-activating sentences. For **embedding** scoring, activating and non-activating sentences are embedded as "documents" that should be retrieved using the explanation as a query.

As an alternative to simulation scoring, we introduce four new evaluation methods that focus on how well an explanation enables a scorer to discriminate between activating and non-activating contexts. As an added benefit, all of these methods are more compute-efficient than simulation.

Currently, there is no consensus over what makes an ideal explanation. In this work, we chose to focus on the idea that the explanation of the feature should accurately distinguish between which contexts the feature is active and which contexts the feature is not active.

3.5.1 Detection

One simple approach for scoring explanations is to ask a language model to identify whether a whole sequence activates a SAE latent given an explanation. Currently, the bottleneck of scoring is on the autoregressive nature of token generation. Detection requires few output tokens for each example used in scoring, meaning examples from a wider distribution of latent activation strengths can be used at the same expense. By including non-activating contexts, this method measures both the precision and recall of the explanation.

Detection is more "forgiving" than simulation insofar as the scorer does not need to localize the feature to a particular token.

Both this method and fuzzing (described next) can leverage token probabilities to estimate how certain the scorer model is of their classification, and this is an effect that we believe can be to improve the scoring methods. Details on the prompt in Appendix A.3.1.

3.5.2 Fuzzing

Fuzzing is similar to detection, but at the level of individual tokens. Here, potentially activating tokens are delimited in each example and the language model is prompted to identify which of the sentences are correctly marked. Evaluating an explanation on both detection and fuzzing can identify whether a model is classifying examples for the correct reason.

While in detection, non-activating examples were used, in fuzzing we choose activating contexts and randomly select non-activating tokens. Details on the prompt on Appendix A.3.2

3.5.3 Surprisal

Surprisal scoring is based on the idea that a good explanation should help a base language model $\mathcal{M}$ (the "scorer") achieve lower cross-entropy loss on activating contexts than it would without a relevant explanation. Specifically, for each context $\mathbf{x}$, activating or non-activating, we measure the *information value* of an explanation $\mathbf{z}$ as $\log p_{\mathcal{M}}(\mathbf{x}|\mathbf{z}) - \log p_{\mathcal{M}}(\mathbf{x}|\tilde{\mathbf{z}})$, where $\tilde{\mathbf{z}}$ is a fixed pseudo-explanation. A good explanation should have higher information value on activating examples than on non-activating ones. The overall surprisal score of the explanation is given by the AUROC of its information value when viewed as a classifier distinguishing activating from non-activating contexts. Details on the prompt and how to compute the score in Appendix A.3.3.

3.5.4 Embedding

We can imagine explanations of latents as "queries" that should be able to retrieve contexts where the latent is active. These contexts can act as "documents" that are embedded by an encoding transformer, and we can use the similarities between the query and the documents to distinguish between activating and non-activating contexts. If the encoding model is small enough, this technique is the fastest and opens up the possibility to evaluate a larger fraction of the activation distribution. We have seen that using a larger embedding model didn't significantly improve the scores, see fig A2, although we believe that this approach was under-investigated. Details on the prompt, on the embedding model and on the way to compute the score in Appendix A.3.4.

3.5.5 Intervention scoring

Unlike the above four context-based scores, intervention scoring interprets a feature's counterfactual impact on model output. We quantify the interpretability of an intervention I with explanation $\mathbf{z}$ on a distribution of prompts π as the average decrease in the scorer's surprisal about the explanation when conditioned on text generated with the intervention.

$$S(I, \mathbf{z}; \pi) = \mathbb{E}_{\mathbf{x}\sim\pi}\Big[\mathbb{E}_{\mathbf{i}\sim\mathcal{G}_I(\mathbf{x})}[\log p_{\mathcal{M}}(\mathbf{z}|\mathbf{i})] - \mathbb{E}_{\mathbf{g}\sim\mathcal{G}(\mathbf{x})}[\log p_{\mathcal{M}}(\mathbf{z}|\mathbf{g})]\Big] \tag{1}$$

$\mathcal{G}(\mathbf{x})$ is the distribution over subject model generations given prompt $\mathbf{x}$ at temperature 1. In $\mathcal{G}_I(\mathbf{x})$, the intervention is applied to the subject model as it generates. In practice we estimate the quantity S by sampling one clean and one intervened generation for each sampled of prompt.

Sufficiently strong interventions can be trivially interpretable by causing the model to deterministically output some logit distribution. Therefore, **interpretability scores of interventions should be compared for interventions of a fixed strength**. We define the strength σ of an arbitrary intervention I as the average KL-divergence of the model's intervened logit distribution with reference to the model's clean output.

$$\sigma(I; \pi) = \mathbb{E}_{\mathbf{x}\sim\pi}\left[D_{KL}(p_{\text{subject}}(\cdot|\mathbf{x}) \mid\mid p_{\text{subject},I}(\cdot|\mathbf{x}))\right] \tag{2}$$

See Appendix A.7 for details on the explanation and scoring pipeline we use in our intervention experiments.

Table 1: Spearman correlation computed over 600 different latent scores

	Fuzzing	Detection	Simulation	Embedding	Surprisal
Fuzzing	1	0.73	0.70	0.41	0.30
Detection		1	0.44	0.71	0.62
Simulation			1	0.33	0.20
Embedding				1	0.79
Surprisal					1

4 RESULTS

4.1 COMPARING SCORING METHODS

Since simulation scoring is an established method, we measure how our other context-based scoring techniques correlate with simulation, as well as how they correlate between themselves (see tables 1 and A1).

We find that fuzzing and detection have the highest Spearman correlations with simulation scoring, at 0.70 and 0.44 respectively. The imperfect correlations hint at either shortcomings of the scoring metrics or the fact that these metrics can measure different qualities of explanations.

Simulation correlates more strongly with fuzzing than with detection, which is expected because fuzzing is similar to simulation scoring. It correlates weakly with surprisal and embedding scoring. On the other hand, surprisal and embedding scores correlate more with detection scoring than with fuzzing, and most strongly with each other. Due to the speed of embedding scoring, we believe it to be a potentially scalable scoring technique for quick iteration.

4.1.1 INTERVENTION SCORING

In figure 4, we see that the intervention score we propose is a valuable contribution to the set of automatic interpretability metrics because it (a) distinguishes between features from a trained SAE and random features and (b) recalls features that context-based scoring methods fail to interpret.

4.2 COMPARING EXPLANATION METHODS

We use a set of 500+ latents as a testbed to measure the effects of design choices and hyperparameters on explanation quality. Each latent is scored using 100 activating and 100 non-activating examples. The activating examples are chosen via stratified sampling such that there are always 10 examples from each of the 10 deciles of the activation distribution.

We evaluate explanation quality using fuzzing, detection and embedding scores, as those were both quick to compute and easy to interpret. We expect this mix of scores to correlate well with simulation scoring and reflect the extent to which the proposed explanation is valid over both activating and non-activating examples.

Table A6 shows that explanations based on more examples tend to have higher scores. By sampling the examples that are shown to the explainer model instead of showing just the top activating examples it is possible to increase the scores of the features, see fig. 3 and A5. This effect would not be seen if the scoring were done on just the most activating examples, underscoring a problem with current auto-interpretability evaluations, which produce explanations using top activating examples and evaluate them on a small subset of the activation distribution.

Increasing the size of the explainer model increases the scores of the explanations but we don't find the performance of Claude Sonnet 3.5 to be much higher than that of Llama 3.1 70b, see A7, and both are similar to those generated by a human. We expect that this may be due to the fact that the prompting techniques were first optimized for Llama 3.1, and that there could be potential gains in optimizing the prompting technique of Claude. Not surprisingly, we also see that even for fuzzing and detection, using a smaller scorer model leads to lower scores.

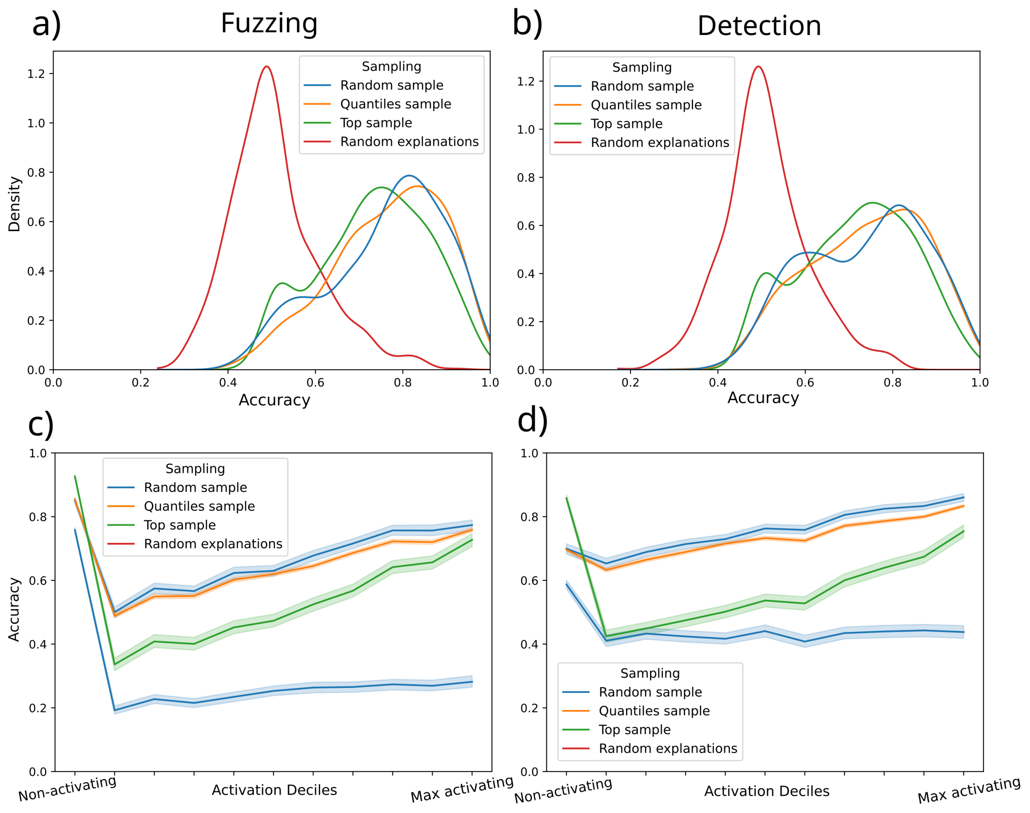


Figure 3: **Fuzzing and detection scores for different sampling techniques.** Panels **a)** and **b)** show the distributions of fuzzing and detection scores, respectively, as a function of different example sampling methods for explanation generation. Sampling only from the top activation gets on average lower accuracy in fuzzing and on detection when compared with randomly sampling and sampling from quantiles. The distributions from random sampling and sampling from quantiles are very similar. Panels **c)** and **d)** measure how the explanations generalize across activation quantiles, showing that explanations generated from the top quantiles are better at distinguishing non-activating examples in detection, but have lower accuracy on other quantiles, especially on the lower activating deciles. This also happens with explanations generated from examples sampled randomly and from quantiles, but the accuracy does not drop as much in lower activating deciles.

In table A4, we don't find a significant dependence on whether we use the same dataset as the training set of the SAEs (see A2) whether we used chain of thought or not (see A3) and whether we change the size of the contexts shown.

4.3 Comparing SAEs

SAEs with more latents have higher scores, and scores that are significantly higher than those of neurons, which are just slightly better than randomly initialized SAEs, see A8. Neurons are more interpretable if made sparser by only considering the top k most activated neurons on a given token, but still significantly underperform SAEs in our tests, see A8.

The location of the SAEs matters; residual stream SAEs have slightly better scores than ones trained on MLP outputs, see A8. We also observe that earlier layers have lower overall scores, but that the scores across model depth remain constant after those layers, see A4.

A detailed analysis of various factors in the auto-interpretability pipeline is in Appendix A.5.

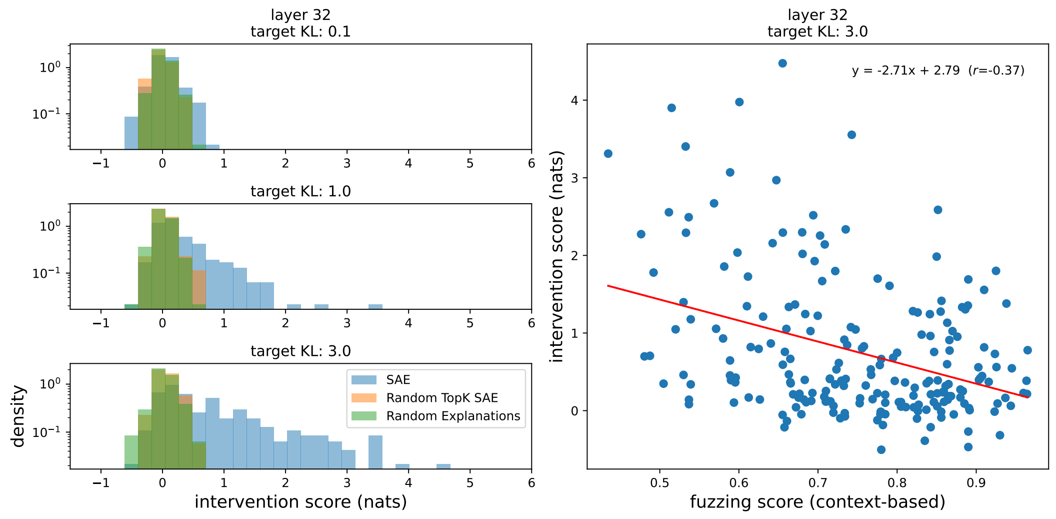


Figure 4: **Intervention scores.** Here we present intervention scores (Sec 3.5.5) for SAE features in Gemma 2 9B at layer 32. **Left:** SAE features are more interpretable than random features, especially when intervening more strongly. Our explainer also produces explanations that are scored higher than random explanations. **Right:** Many features that would normally be uninterpreted when using context-based automatic interpretability are interpretable in terms of their effects on output.

5 Overlap between latents at adjacent layers

Because each block in a transformer performs an incremental additive update to the residual stream, one would expect SAEs trained at nearby layers to learn similar features. Measuring the degree of feature overlap is interesting for a few reasons. First, if adjacent SAEs learn almost identical features, it may not be worthwhile to train and interpret SAEs at every single residual stream layer. Second, we can use feature overlap to sanity check our auto interpretability pipeline: if the explanation for latent α at layer j is very different from the explanation for "the same" latent at layer $j + 1$, this would suggest that our pipeline is inconsistent and noisy. Finally, feature overlap may allow us to estimate the degree of *semantic* similarity between layers, as opposed to mere statistical similarity.

Unfortunately, we cannot simply fix some index i and compare latent i at layer j to latent i at layer $j+1$. This is due to permutation symmetry: given an SAE with parameters $(\mathbf{W}_e, \mathbf{b}_e, \mathbf{W}_d, \mathbf{b}_d)$ and a permutation matrix $\mathbf{P} \neq \mathbf{I}$, the SAE $(\mathbf{P}\mathbf{W}_e, \mathbf{P}\mathbf{b}_e, \mathbf{P}^T\mathbf{W}_d, \mathbf{P}^T\mathbf{b}_d)$ has identical input-output behavior, and yet its latents are completely "unaligned" with the original. Instead, we use the Hungarian algorithm (Crouse, 2016) to compute the permutation which maximizes the Frobenius inner product between the decoder weight matrices of the two SAEs. A similar method was used by Ainsworth et al. (2023) to align weights of independently trained MLPs.

In this section, we focus on the 16k latent SAEs trained on Gemma 2 9b. In Appendix A.6, we report the pairwise Frobenius inner products between SAE decoder matrices at different layers before and after aligning them with the Hungarian algorithm. As expected, this simple metric shows that SAEs trained on adjacent layers of the residual stream are more similar than those trained on the MLPs.

To measure the semantic similarity between the explanations, we embed the explanation of each latent and organize the embeddings into a matrix, sorted with the indices found by the Hungarian algorithm. We then compute the Frobenius inner product of these matrices, ignoring the latents that don't have an explanation on both layers. We observe that the residual stream SAEs have higher semantic overlap for neighboring layers than the MLP SAEs.

Our results suggest that, when training SAEs given a limited compute budget, one should prioritize training *wider* SAEs on a *smaller* subset of residual stream layers, perhaps every k^{th} layer for some stride $k > 1$. If feature diversity is a priority, it may make more sense to train SAEs on MLP outputs exclusively, although we find MLPs to be slightly less interpretable than the residual stream, see A8.

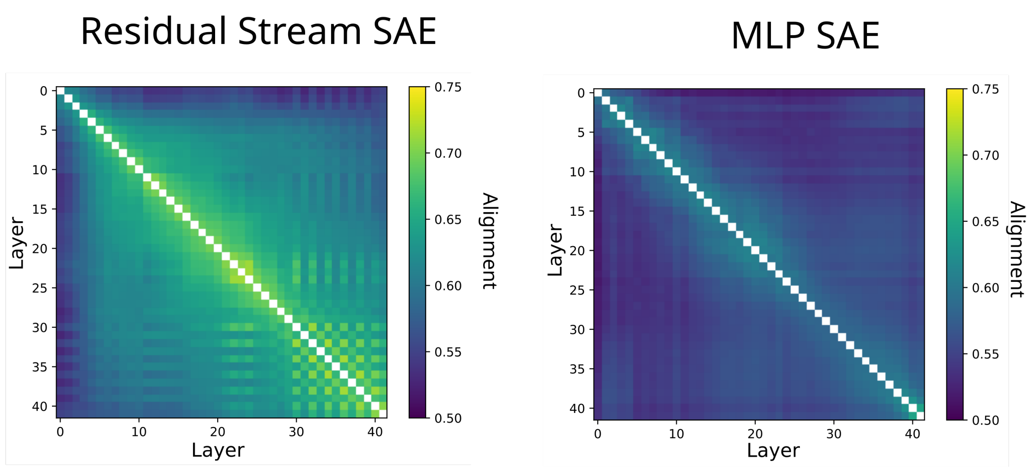


Figure 5: **Alignment between embeddings of explanations of the 16k Gemma 2 9b SAEs.** After "aligning" the features of each layers by using the decoder weights, we observe that the explanations found for the latents of the residual stream SAE are more similar than for latents of the MLPs. In this figure, alignment is the measured by the Frobenius inner product between the feature embeddings.

It's worth noting that the Hungarian algorithm could also be used to align latents from independently trained SAEs in other contexts, such as SAEs trained on different checkpoints of the same training run, or even completely different architectures trained on the same data (Moschella et al., 2022). We leave these extensions to future work.[3]

6 Conclusion

Explaining the latents of SAEs trained on cutting-edge LLMs is a computationally demanding task, requiring scalable methods to both generate explanations and assess their quality. We addressed issues with the conventional simulation-based scoring and introduced four new scoring techniques, each with distinct strengths and limitations. These methods allowed us to explore the "prompt design space" for generating effective explanations and propose practical guidelines.

While most latents are well-explained by contexts of length 32, we expect features active over longer contexts to be important and plan to develop better methods for capturing these long-context features. Additionally, although current scoring methods do not account for explanation length, we believe shorter explanations are generally more useful and will incorporate this in future metrics. Some scoring methods also require further refinement, particularly in selecting non-activating examples to improve evaluation.

By analyzing a large number of explanations, we examined overlaps across layers and the distribution of explanations in decoder direction space. Our results suggest that, when compute resources are constrained, it may be more efficient to train wider SAEs on a small subset of residual stream layers, rather than narrower SAEs on all layers.

Access to better, automatically generated explanations could play a crucial role in areas like model steering, concept localization, and editing. We hope that our efficient scoring techniques will enable feedback loops to further enhance the quality of explanations.

References

Samuel Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. Git re-basin: Merging models modulo permutation symmetries. In *The Eleventh International Conference on Learning Representa-*

[3]All that is needed is an adequate cost function for measuring the similarity of individual latents. SciPy's implementation supports rectangular cost matrices, allowing SAEs of different sizes to be (partially) aligned.

tions, 2023. URL https://openreview.net/forum?id=CQsmMYmlP5T.

Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. Linear algebraic structure of word senses, with applications to polysemy. *Transactions of the Association for Computational Linguistics*, 6:483–495, 2018.

Anton Bakhtin, Noam Brown, Emily Dinan, Gabriele Farina, Colin Flaherty, Daniel Fried, Andrew Goff, Jonathan Gray, Hengyuan Hu, Athul Paul Jacob, Mojtaba Komeili, Karthik Konath, Minae Kwon, Adam Lerer, Mike Lewis, Alexander H. Miller, Sandra Mitts, Adithya Renduchintala, Stephen Roller, Dirk Rowe, Weiyan Shi, Joe Spisak, Alexander Wei, David J. Wu, Hugh Zhang, and Markus Zijlstra. Human-level play in the game of diplomacy by combining language models with strategic reasoning. *Science*, 378:1067 – 1074, 2022. URL https://api.semanticscholar.org/CorpusID:253759631.

Leonard Bereska and Efstratios Gavves. Mechanistic interpretability for ai safety–a review. *arXiv preprint arXiv:2404.14082*, 2024.

Steven Bills, Nick Cammarata, Dan Mossing, Henk Tillman, Leo Gao, Gabriel Goh, Ilya Sutskever, Jan Leike, Jeff Wu, and William Saunders. Language models can explain neurons in language models. *URL https://openaipublic. blob. core. windows. net/neuron-explainer/paper/index. html.(Date accessed: 14.05. 2023)*, 2, 2023.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Peter Bühlmann. Invariance, causality and robustness. *arXiv preprint arXiv:1812.08233*, 2018.

Haozhe Chen, Carl Vondrick, and Chengzhi Mao. Selfie: Self-interpretation of large language model embeddings, 2024. URL https://arxiv.org/abs/2403.10949.

Together Computer. Redpajama: an open dataset for training large language models, 2023. URL https://github.com/togethercomputer/RedPajama-Data.

David F Crouse. On implementing 2d rectangular assignment algorithms. *IEEE Transactions on Aerospace and Electronic Systems*, 52(4):1679–1696, 2016.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. *arXiv preprint arXiv:2309.08600*, 2023.

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.

Asma Ghandeharioun, Avi Caciularu, Adam Pearce, Lucas Dixon, and Mor Geva. Patchscopes: A unifying framework for inspecting hidden representations of language models, 2024. URL https://arxiv.org/abs/2401.06102.

Wes Gurnee, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris Bertsimas. Finding neurons in a haystack: Case studies with sparse probing. *arXiv preprint arXiv:2305.01610*, 2023.

Wes Gurnee, Theo Horsley, Zifan Carl Guo, Tara Rezaei Kheirkhah, Qinyi Sun, Will Hathaway, Neel Nanda, and Dimitris Bertsimas. Universal neurons in gpt2 language models. *arXiv preprint arXiv:2401.12181*, 2024.

Evan Hernandez, Sarah Schwettmann, David Bau, Teona Bagashvili, Antonio Torralba, and Jacob Andreas. Natural language descriptions of deep visual features, 2022. URL https://arxiv.org/abs/2201.11114.

Jing Huang, Atticus Geiger, Karel D'Oosterlinck, Zhengxuan Wu, and Christopher Potts. Rigorously assessing natural language explanations of neurons. *arXiv preprint arXiv:2309.10312*, 2023.

Caden Juang, Gonçalo Paulo, Jacob Drori, and Belrosem Nora. Understanding and steering Llama 3, 9 2024. URL https://goodfire.ai/blog/research-preview/.

Dmitrii Kharlapenko, neverix, Neel Nanda, and Arthur Conmy. Self-explaining SAE features, 8 2024. URL https://www.lesswrong.com/posts/8ev6coxChSWcxCDy8/self-explaining-sae-features.

Laura Kopf, Philine Lou Bommer, Anna Hedström, Sebastian Lapuschkin, Marina M. C. Höhne, and Kirill Bykov. Cosy: Evaluating textual explanations of neurons, 2024. URL https://arxiv.org/abs/2405.20331.

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, 2024. URL https://arxiv.org/abs/2408.05147.

Johnny Lin and Joseph Bloom. Neuronpedia: Interactive reference and tooling for analyzing neural networks with sparse autoencoders, 2023. URL https://www.neuronpedia.org. Software available from neuronpedia.org.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The ai scientist: Towards fully automated open-ended scientific discovery, 2024. URL https://arxiv.org/abs/2408.06292.

Tom Macgrath. Understanding and steering Llama 3, 9 2024. URL https://goodfire.ai/blog/research-preview/.

Luca Moschella, Valentino Maiorca, Marco Fumero, Antonio Norelli, Francesco Locatello, and Emanuele Rodolà. Relative representations enable zero-shot latent space communication. *arXiv preprint arXiv:2209.15430*, 2022.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. Mteb: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316*, 2022. doi: 10.48550/ARXIV.2210.07316. URL https://arxiv.org/abs/2210.07316.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. *Distill*, 5(3), March 2020. ISSN 2476-0757. doi: 10.23915/distill.00024.001. URL http://dx.doi.org/10.23915/distill.00024.001.

OpenAI. Gpt-4 technical report, 2023. URL https://arxiv.org/abs/2303.08774.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. *ArXiv*, abs/2311.03658, 2023. URL https://api.semanticscholar.org/CorpusID:265042984.

Bernhard Schölkopf, Dominik Janzing, Jonas Peters, Eleni Sgouritsa, Kun Zhang, and Joris Mooij. On causal and anticausal learning. *arXiv preprint arXiv:1206.6471*, 2012.

Tamar Rott Shaham, Sarah Schwettmann, Franklin Wang, Achyuta Rajaram, Evan Hernandez, Jacob Andreas, and Antonio Torralba. A multimodal automated interpretability agent, 2024. URL https://arxiv.org/abs/2404.14394.

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, 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. *Transformer Circuits Thread*, 2024. URL https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models, 2023. URL https://arxiv.org/abs/2305.16291.

A APPENDIX

A.1 EXPLAINER PROMPT

The system prompt if not using Chain of Thought (COT):

```
You are a meticulous AI researcher conducting an important
investigation into patterns found in language. Your task is to
analyze text and provide an explanation that thoroughly
encapsulates possible patterns found in it.
Guidelines:

You will be given a list of text examples on which special words
are selected and between delimiters like << this >>.
If a sequence of consecutive tokens all are important,
the entire sequence of tokens will be contained between
delimiters <<just like this>>. How important each token is for
the behavior is listed after each example in parentheses.

- Try to produce a concise final description. Simply describe
the text features that are common in the examples, and what
patterns you found.

- If the examples are uninformative, you don't need to mention
them. Don't focus on giving examples of important tokens,
but try to summarize the patterns found in the examples.

- Do not mention the marker tokens ($<<$ $>>$)
in your explanation.

- Do not make lists of possible explanations.
Keep your explanations short and concise.

- The last line of your response must be the formatted
explanation, using [EXPLANATION]:
```

We add to the previous prompt the following if we want to do COT:

```
To better find the explanation for the language patterns,
go through the following stages:

1.Find the special words that are selected in the examples
and list a couple of them. Search for patterns in these words,
if there are any. Don't list more than 5 words.

2. Write down general shared features of the text examples.
This could be related to the full sentence or to the words
surrounding the marked words.

3. Formulate a hypothesis and write down the final explanation
using [EXPLANATION]:.
```

One of the few shot examples of how examples are displayed to the model.

```
Example 1:  and he was <<over the moon>> to find

Activations: (``over", 5), (`` the", 6), (`` moon", 9)

Example 2:  we'll be laughing <<till the cows come home>>! Pro
```

```
Activations: (``till", 5), (`` the", 5),
(`` cows", 8), (`` come", 8),
(`` home", 8)

Example 3:  thought Scotland was boring, but really there’s
more <<than meets the eye>>! I’d

Activations: (``than", 5), (`` meets", 7), (`` the", 6),
(`` eye", 8)
```

If COT is used, an explicit example of using COT is demonstrated.

```
ACTIVATING TOKENS: ``over the moon", ``than meets the eye".
SURROUNDING TOKENS: No interesting patterns.

Step 1.
- The activating tokens are all parts of common idioms.
- The surrounding tokens have nothing in common.

Step 2.
- The examples contain common idioms.
- In some examples, the activating tokens are followed
by an exclamation mark.

Step 3.
- The activation values are the highest for the more common
idioms in examples 1 and 3.

Let me think carefully. Did I miss any patterns in the text
examples? Are there any more linguistic similarities?

- Yes, I missed one: The text examples all convey positive
sentiment.
```

Afterwards an explanation is added to the example

```
[EXPLANATION]: Common idioms in text conveying positive
sentiment.
```

A.2 Examples of activating contexts and explanations

As discussed in the main text, the explanations found for a given latent can be very different depending on the way to sample the activating contexts shown to the explainer model, see fig. A1. This has its advantages and disadvantages.

When sampling from only the top activations, it is possible that the explainer model gives an explanation that is more narrow - “The concept of a buffer, referring to something that separates, shields, or protects one thing from another, often used in various contexts such as physical barriers, chemical reactions, or digital data processing” - instead of one that captures the full distribution - “Words or phrases associated with concepts of spatial or temporal separation (buffers, zones, or cushions) or colloquialisms (buff, buffs, or Buffy, referring to a popular TV show or enthusiast)”. Here the narrower explanations resulted in lower scores: the explanation generated from randomly sampled examples scored 0.95 accuracy in fuzzing and 0.93 accuracy in detection, while the explanation generated from top examples only achieves 0.8 accuracy in fuzzing and 0.74 accuracy in detection.

On the other hand, if we look at the second example, sampling from random examples may sometimes confuse the explainer model - “Tokens often precede or succeed prepositions, articles, and words that signal possession or quantity, frequently indicating a relationship between objects or actions. of feature” , which might not see the pattern that is more clear in the top activating examples

-“Descriptions of food or events where food is involved, often mentioning leftovers, and sometimes mentioning the act of eating, serving, or storing food, as well as the amount of food, or the pleasure or satisfaction derived from it.”. Here we see very poor scores on the explanation from randomly sampled examples, 0.54 accuracy in detection and 0.57 accuracy in fuzzing, while the explanation generated from the top examples has 0.89 accuracy both in detection and fuzzing (over the full distribution).

A.3 DIFFERENT SCORING METHODS

A.3.1 DETECTION DETAILS

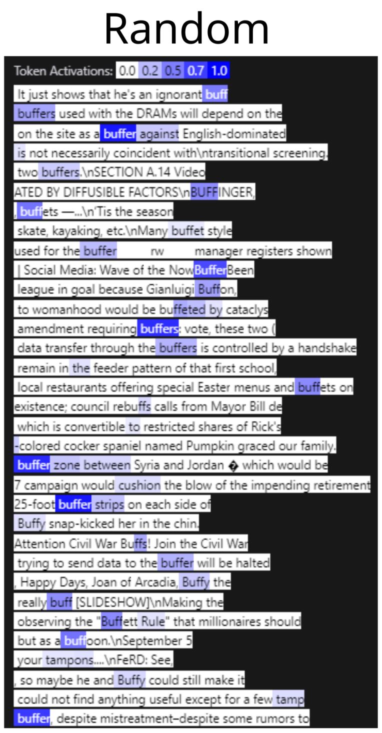
The prompt used for detection scoring is the following:

```
You are an intelligent and meticulous linguistics researcher.

You will be given a certain feature of text, such as
``male pronouns" or ``text with negative sentiment".

You will then be given several text examples. Your task
is to determine which examples possess the feature.

For each example in turn, return 1 if the sentence is
correctly labeled or 0 if the tokens are mislabeled. You must
return your response in a valid Python list. Do not return
anything else besides a Python list.
```

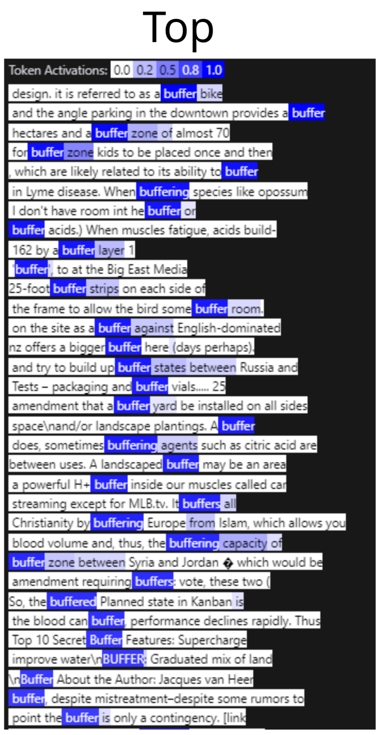
Together with the prompt, there are several few shot examples like the following:

```
<user_prompt>

Feature explanation: Words related to American football
positions, specifically the tight end position.

Text examples:

Example 0:Getty Images Patriots tight end Rob Gronkowski
had his boss

Example 1: names of months used in The Lord of the Rings:
the

Example 2: Media Day 2015 LSU defensive end Isaiah Washington
(94) speaks to the

Example 3: shown, is generally not eligible for ads. For
example, videos about recent tragedies,

Example 4: line, with the left side namely tackle Byron
Bell at tackle and guard Amini

<assistant_response>

[1,0,0,0,1]
```

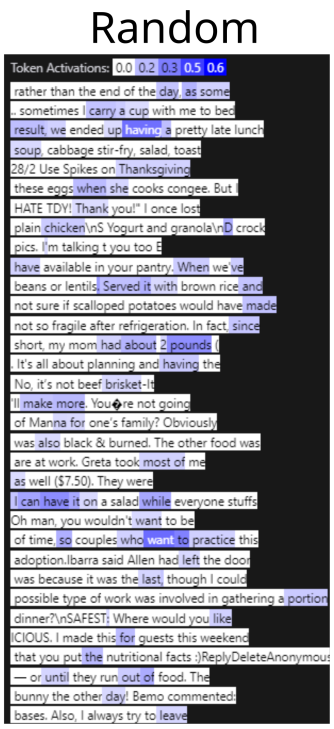
In this example the <user_prompt> and the <assistant_response> tags are substituted with the correct instruct format used by the scorer model. In detection scoring, 5 shuffled examples are shown to the model at the same time.



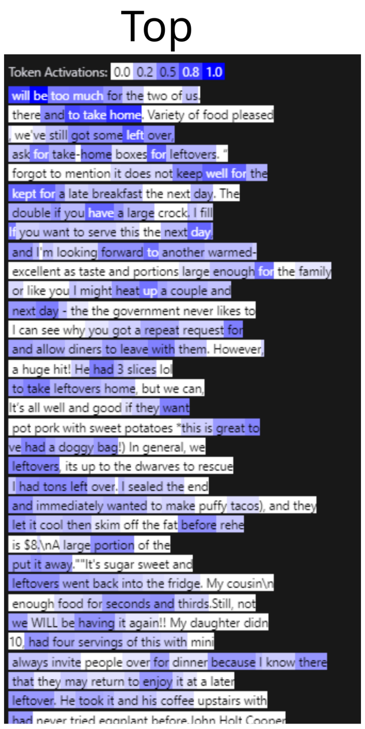
```
It just shows that he's an ignorant buff
buffers used with the DRAMs will depend on the
on the site as a buffer against English-dominated
is not necessarily coincident with\ntransitional screening.
two buffers.\nSECTION A.14 Video
ATED BY DIFFUSIBLE FACTORS\nBUFFINGER,
, buffets —...\n'Tis the season
skate, kayaking, etc.\nMany buffet style
used for the buffer        rw        manager registers shown
| Social Media: Wave of the NowBufferBeen
league in goal because Gianluigi Buffon,
to womanhood would be buffeted by cataclys
amendment requiring buffers: vote, these two (
data transfer through the buffers is controlled by a handshake
remain in the feeder pattern of that first school,
local restaurants offering special Easter menus and buffets on
existence; council rebuffs calls from Mayor Bill de
which is convertible to restricted shares of Rick's
-colored cocker spaniel named Pumpkin graced our family.
buffer zone between Syria and Jordan � which would be
7 campaign would cushion the blow of the impending retirement
25-foot buffer strips on each side of
Buffy snap-kicked her in the chin.
Attention Civil War Buffs! Join the Civil War
trying to send data to the buffer will be halted
, Happy Days, Joan of Arcadia, Buffy the
really buff [SLIDESHOW]\nMaking the
observing the "Buffett Rule" that millionaires should
but as a buffoon.\nSeptember 5
your tampons....\nFeRD: See,
, so maybe he and Buffy could still make it
could not find anything useful except for a few tamp
buffer, despite mistreatment–despite some rumors to
```



```
design. it is referred to as a buffer bike
and the angle parking in the downtown provides a buffer
hectares and a buffer zone of almost 70
for buffer zone kids to be placed once and then
, which are likely related to its ability to buffer
in Lyme disease. When buffering species like opossum
I don't have room int he buffer or
buffer acids.) When muscles fatigue, acids build-
162 by a buffer layer 1
'buffer', to at the Big East Media
25-foot buffer strips on each side of
the frame to allow the bird some buffer room.
on the site as a buffer against English-dominated
nz offers a bigger buffer here (days perhaps).
and try to build up buffer states between Russia and
Tests – packaging and buffer vials..... 25
amendment that a buffer yard be installed on all sides
space\nand/or landscape plantings. A buffer
does, sometimes buffering agents such as citric acid are
between uses. A landscaped buffer may be an area
a powerful H+ buffer inside our muscles called car
streaming except for MLB.tv. It buffers all
Christianity by buffering Europe from Islam, which allows you
blood volume and, thus, the buffering capacity of
buffer zone between Syria and Jordan � which would be
amendment requiring buffers: vote, these two (
So, the buffered Planned state in Kanban is
the blood can buffer, performance declines rapidly. Thus
Top 10 Secret Buffer Features: Supercharge
improve water\nBUFFER: Graduated mix of land
\nBuffer About the Author: Jacques van Heer
buffer, despite mistreatment–despite some rumors to
point the buffer is only a contingency. [link
```



```
rather than the end of the day, as some
.. sometimes I carry a cup with me to bed
result, we ended up having a pretty late lunch
soup, cabbage stir-fry, salad, toast
28/2 Use Spikes on Thanksgiving
these eggs when she cooks congee. But I
HATE TDY! Thank you!" I once lost
plain chicken\nS Yogurt and granola\nD crock
pics. I'm talking t you too E
have available in your pantry. When we've
beans or lentils. Served it with brown rice and
not sure if scalloped potatoes would have made
not so fragile after refrigeration. In fact, since
short, my mom had about 2 pounds (
. It's all about planning and having the
No, it's not beef brisket-It
'll make more. You�re not going
of Manna for one's family? Obviously
was also black & burned. The other food was
are at work. Greta took most of me
as well ($7.50). They were
I can have it on a salad while everyone stuffs
Oh man, you wouldn't want to be
of time, so couples who want to practice this
adoption.Ibarra said Allen had left the door
was because it was the last, though I could
possible type of work was involved in gathering a portion
dinner?\nSAFEST: Where would you like
ICIOUS. I made this for guests this weekend
that you put the nutritional facts :)ReplyDeleteAnonymous
— or until they run out of food. The
bunny the other day! Bemo commented:
bases. Also, I always try to leave
```



```
will be too much for the two of us.
there and to take home. Variety of food pleased
, we've still got some left over,
ask for take-home boxes for leftovers. "
forgot to mention it does not keep well for the
kept for a late breakfast the next day. The
double if you have a large crock. I fill
If you want to serve this the next day,
and I'm looking forward to another warmed-
excellent as taste and portions large enough for the family
or like you I might heat up a couple and
next day - the the government never likes to
I can see why you got a repeat request for
and allow diners to leave with them. However,
a huge hit! He had 3 slices lol
to take leftovers home, but we can,
It's all well and good if they want
pot pork with sweet potatoes *this is great to
ve had a doggy bag!) In general, we
leftovers, its up to the dwarves to rescue
I had tons left over. I sealed the end
and immediately wanted to make puffy tacos), and they
let it cool then skim off the fat before rehe
is $8.\nA large portion of the
put it away.""It's sugar sweet and
leftovers went back into the fridge. My cousin\n
enough food for seconds and thirds.Still, not
we WILL be having it again!! My daughter didn
10, had four servings of this with mini
always invite people over for dinner because I know there
that they may return to enjoy it at a later
leftover. He took it and his coffee upstairs with
had never tried eggplant before.John Holt Cooper
```

Figure A1: Activating contexts of feature 209 (top) and 293 (bottom), from layers 8 and 32, respectively, of the 131k latent SAE trained on the residual stream of Gemma 2 9b. The shown examples are similar to the ones given to the explainer model to come up with explanations.

A.3.2 FUZZING DETAILS

The prompt used for fuzzing scoring is the following:

```
You are an intelligent and meticulous linguistics researcher.

You will be given a certain feature of text, such as ``male
pronouns" or ``text with negative sentiment". You will
be given a few examples of text that contain this feature.
Portions of the sentence which strongly represent this
feature are between tokens << and >>.

Some examples might be mislabeled. Your task is to determine
if every single token within << and >> is correctly
labeled. Consider that all provided examples could be correct,
none of the examples could be correct, or a mix. An example
is only correct if every marked token is representative
of the feature

For each example in turn, return 1 if the sentence is correctly
labeled or 0 if the tokens are mislabeled. You must return
your response in a valid Python list. Do not return anything
else besides a Python list.
```

Followed by few-shot examples of the kind:

```
<user_prompt>

Feature explanation: Words related to American football
positions, specifically the tight end position.

Text examples:

Example 0:Getty Images Patriots<< tight end>> Rob Gronkowski
had his boss

Example 1: posted You should know this<< about>> offensive
line coaches: they are large, demanding<< men>>

Example 2: Media Day 2015 LSU<< defensive>> end Isaiah
Washington (94) speaks<< to the>>

Example 3:<< running backs>>,`` he said. .. Defensive
<< end>>
Carroll Phillips is improving and his injury is

Example 4:<< line>>, with the left side namely<< tackle>>
Byron Bell at<< tackle>> and<< guard>> Amini

<assistant_response>

[1,0,0,1,1]
```

In this example, the <user_prompt> and the <assistant_response> tags are substituted with the correct instruct format used by the scorer model. In fuzzing scoring, 5 shuffled examples are shown to the model at the same time.

A.3.3 SURPRISAL DETAILS

In surprisal scoring, the cross-entropy loss of the scorer model is computed over the tokens of an example. For each explanation, this loss is computed with the explanation and with a default explanation - “Various unrelated sentences,” where the examples can either be activating context or non-activating contexts. In our first approach, Llama 3.1 70b base was used. The prompt starts with few shot examples like:

```
The following is a description of a certain feature of text
and a list of examples that contain the feature.

Description:

References to the Antichrist, the Apocalypse and conspiracy
theories related to those topics.

Sentences:

 `` by which he distinguishes Antichrist is, that he would
 rob God of his honour and take it to himself, he gives
 the leading feature which we ought  "

 ``3 begins. And the rise of Antichrist. Get ready with   "

 `` would be destroyed. The worlds economy would likely
 collapse as a result and could usher in a one world government
 movement. I wrote a small 6 page  "

Description:

Sentences containing digits forming a four-digit year.

Sentences:

 `` 20, 2013 at 7:41 pm Martin Smith  "

 `` of 2012. In other words, Italy's   "

 ``end 2012 levels). In the first quarter of 2013, we expect
 revenue to be up slightly from the fourth quarter  "

Description:

Text related to banking and financial institutions.

Sentences:

 ``: He is on the Board of Directors with the Lumbee Bank   "

 `` refurbishing the Bank's branches. BIP reached 400
 thousand users in one year The use of BIP has already
 doubled The  "

 `` the Federal Deposit Insurance Corp.  "

Description:

Occurrences of the word 'The' at the beginning of sentence.
```

```
Sentences:

 ``The Smoking Tire hits the canyons with one of the fastest
 Audi's on the road  "

 ``The Chairman of the ABI  "

 ``The administrative center is the town of Koch.  "
```

With the pairs of losses with the explanation and with the default explanation, it is possible to compute the decrease in loss caused by having access to the explanation. It is expected that in activating contexts this difference will be greater than in non-activating examples, and the score of the explanation is given by the AUC computed using this loss as a proxy for activating and non-activating labels.

A.3.4 EMBEDDING DETAILS

For embedding scoring, we use a small 400M parameter model. We chose a small performant model on MTEB (Muennighoff et al., 2022) because we found similar scores when using a larger 7B parameter model, see fig A2, as this size allowed us to do increase the number of examples used in scoring.

A set of activating and randomly selected non-activating examples are embedded using the scorer model. Then a "query" instruction is embedded:

```
Instruct: Retrieve sentences that could be related to
the explanation. Query:  \{explanation\}
```

The cosine-similarity between the instruction embedding and the examples is computed and is used as a proxy for activating and non-activating labels when computing the AUC, which is the score of that explanation.

A.3.5 ADVERSARIAL EXAMPLES

Most latents are active in less than 0.1% of the full dataset used to collect the activations, making random non-activating contexts very diverse. Randomly sampling non-activating examples cannot be used to determine whether a latent fires in a token in a specific context or on that token in general, as it is unlikely that that token randomly occurs in each non-activating example. As SAEs are scaled and latents become sparser and more specific, techniques that overly rely on activating contexts will have more imprecise results Gao et al. (2024).

Motivated by the phenomenon of feature splitting, we could use "similar" latents to test whether explanations are precise enough to distinguish between similar contexts. A potential approach is using cosine similarity between decoder directions of latents to find counterexamples for an explanation. Some works (Juang et al., 2024; Macgrath, 2024) have shown that a significant fraction of current latent explanations can't be used to distinguish between features with high similarity.

A.3.6 CORRELATION BETWEEN SCORES

We compute the correlation between these different scores and simulation 600 different latent explanations, of the 131k latent SAE trained on the residual stream of Gemma 2 9b spread across 4 different layers.

A.4 SAE MODELS USED

Throughout this work, we used different SAEs trained on Gemma. The 16k latent ones trained on the MLP have the following average L0 norms per layer:

0:50, 1:56, 2:33, 3:55, 4:66, 5:46, 6:46, 7:47, 8:55, 9:40, 10:49, 11:34, 12:42, 13:40, 14:41, 15:45, 16:37, 17:41, 18:36, 19:38, 20:41, 21:34, 22:34, 23:73, 24:32, 25:72, 26:57, 27:52,

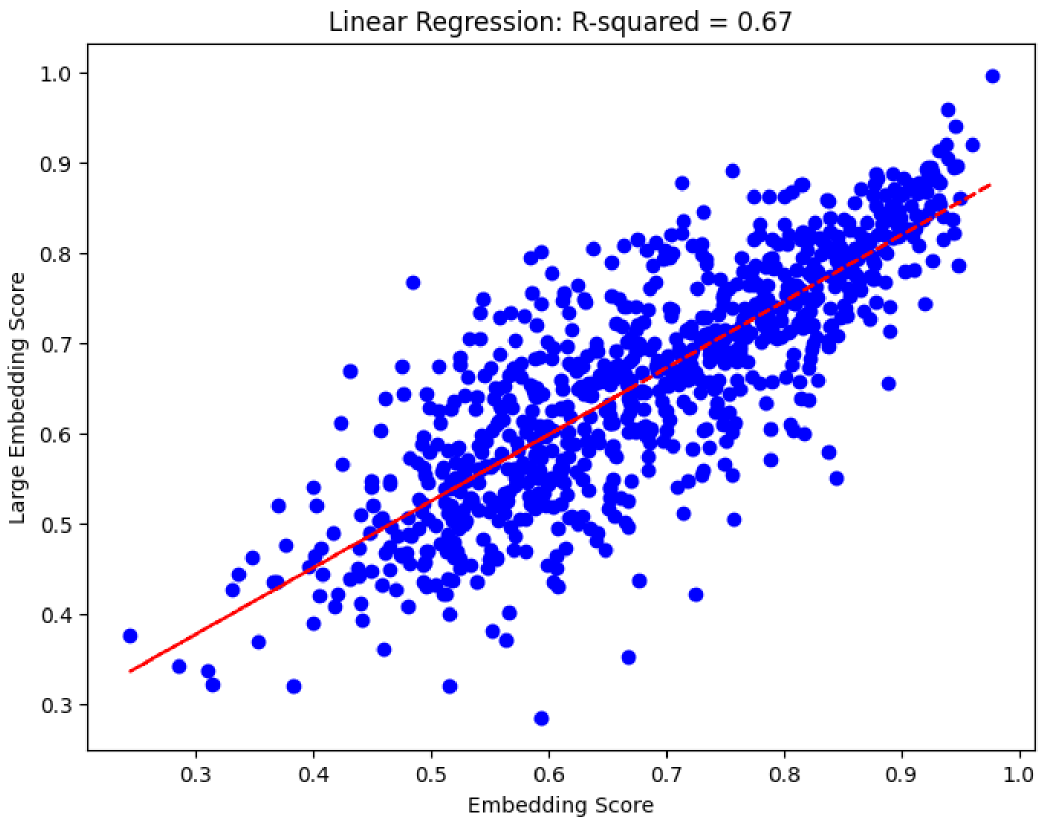


Figure A2: **Comparison between the scores given by a small embedding model and a larger one**

Table A1: Pearson correlation computed over 600 different latent scores

	Fuzzing	Detection	Simulation	Embedding	Surprisal
Fuzzing	1	0.74	0.70	0.42	0.32
Detection		1	0.45	0.70	0.62
Simulation			1	0.34	0.19
Embedding				1	0.79
Surprisal					1

28:50, 29:49, 30:51, 31:43, 32:44, 33:48, 34:47, 35:46, 36:47, 37:53, 38:45, 39:43, 40:37, 41:58

The 16k latent ones trained on the residual stream have the following average L0 norms:

0:35, 1:69, 2:67, 3:37, 4:37, 5:37, 6:47, 7:46, 8:51, 9:51, 10:57, 11:32, 12:33, 13:34, 14:35, 15:34, 16:39, 17:38, 18:37, 19:35, 20: 36, 21:36, 22: 35, 23: 35, 24: 34, 25: 34, 26: 35, 27:36, 28: 37, 29:38, 30:37, 31:35, 32: 34, 33:34, 34:34, 35:34, 36:34, 37:34, 38:34, 39:34, 40:32, 41:52

The 131k latent ones trained in the residual steam have the following average L0 norms:

0:30, 1:33, 2:36, 3:46, 4:51, 5:51, 6:66, 7:38, 8:41, 9:42, 10:47, 11:49, 12:52, 13:30, 14:56, 15:55, 16:35, 17:35, 18:34, 19:32, 20:34, 21:33, 22:32, 23:32, 24: 55, 25:54, 26:32, 27:33, 28: 32, 29:33, 30:32, 31:52, 32: 51, 33:51, 34:51, 35:51, 36: 51, 37:53, 38:53, 39:54, 40: 49, 41:45

We also used SAEs with 65k latents, trained on the residual stream and the MLP of Llama 8b.

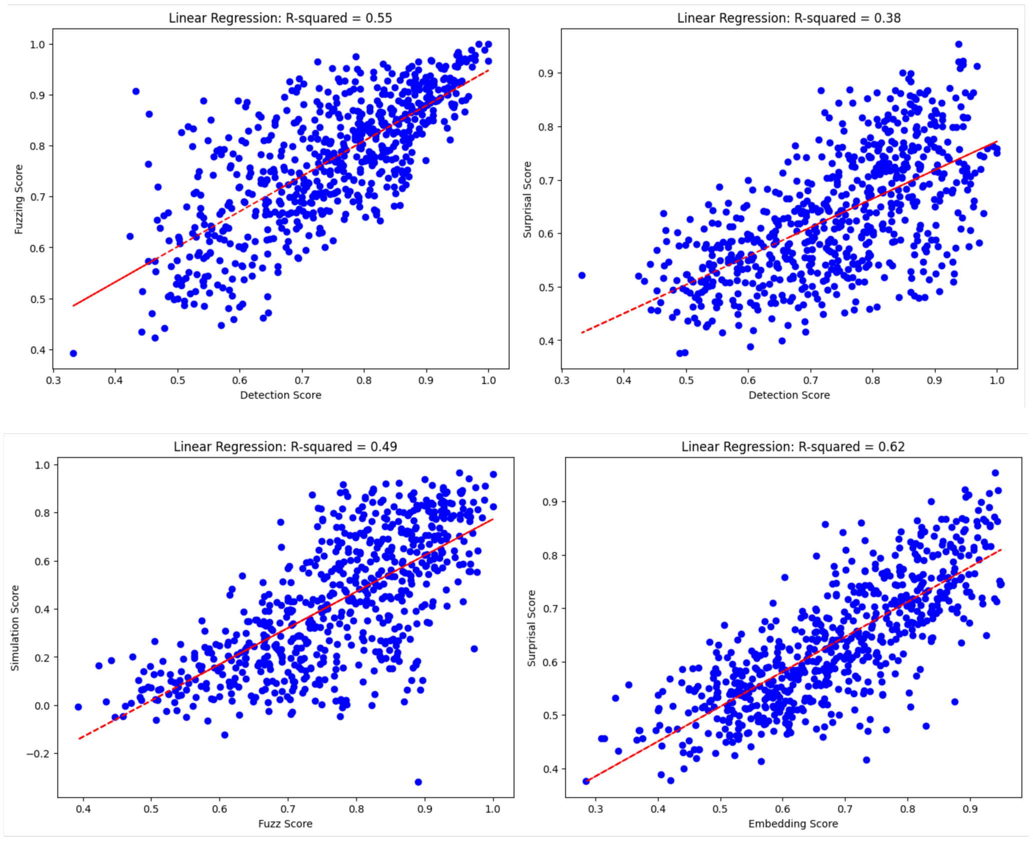


Figure A3: **Scatter plots with different combinations of scores**

Table A2: Impact of training dataset on Fuzzing and Detection performance. Numbers shown are the median score and the interquartile (25%-75%) range.

Experiment	Fuzzing	Detection	Embedding
Random explanation	0.51 (0.45–0.57)	0.51 (0.45–0.58)	0.51 (0.44–0.57)
Randomly initialized Topk SAE	0.55 (0.50–0.60)	0.54 (0.50–0.59)	–
RPJ-v2	0.76 (0.67–0.86)	0.74 (0.63–0.85)	0.67 (0.57–0.80)
Pile	**0.76 (0.67–0.86)**	**0.76 (0.67–0.85)**	**0.69 (0.57–0.80)**

A.5 Factors that influence the explainability

A.5.1 Dependence on dataset

Even though a significant portion of SAE latents being less active when using RPJv2 instead of the Pile, we find that the latents that are active are generally interpretable to the same degree. These evaluations were done using the 131k latent SAE trained on the residual stream of Gemma 2 9b. The scorer and the explainer model where Llama 3.1b 70b instruct, quantized to 4bit.

A.5.2 Dependence on chain of thought and activation information

We find that COT slightly increases the scores of the explanations found, and that it significantly slows down the rate at which one can produce explanations. Giving the explainer model, the activations associated with each token seem to slightly increase the scores of the explanations generated.

Table A3: Impact of prompt content on Fuzzing and Detection performance. Numbers shown are the median score and the interquartile (25%-75%) range.

Experiment	Fuzzing	Detection	Embedding
Random explanation	0.51 (0.45–0.57)	0.51 (0.45–0.58)	0.51 (0.44–0.57)
Randomly initialized Topk SAE	0.55 (0.50–0.60)	0.54 (0.50–0.59)	–
Activations in prompt	**0.76 (0.67–0.86)**	**0.74 (0.63–0.85)**	**0.68 (0.57–0.80)**
No activations in prompt	0.75 (0.65–0.86)	0.73 (0.60–0.84)	0.68 (0.58–0.79)
COT in prompt	0.76 (0.67–0.86)	0.73 (0.61–0.85)	0.65 (0.55–0.75)

Table A4: Impact of context length on Fuzzing and Detection performance. Numbers shown are the median score and the interquartile (25%-75%) range.

Experiment	Fuzzing	Detection	Embedding
Random explanation	0.51 (0.45–0.57)	0.51 (0.45–0.58)	0.51 (0.44–0.57)
Randomly initialized Topk SAE	0.55 (0.50–0.60)	0.54 (0.50–0.59)	–
16 context	0.75 (0.65–0.86)	0.74 (0.62–0.85)	0.70 (0.59–0.81)
32 context	**0.76 (0.67–0.86)**	**0.74 (0.63–0.85)**	0.67 (0.57–0.80)
64 context	0.74 (0.64–0.64)	0.70 (0.57–0.81)	0.65 (0.54–0.78)

These evaluations were done using the 131k latent SAE trained on the residual stream of Gemma 2 9b. The scorer and the explainer model where Llama 3.1b 70b instruct, quantized to 4bit.

A.5.3 Dependence on context length

The context length of the shown examples did not significantly change the scores obtained for the explanations. These evaluations were done using the 131k latent SAE trained on the residual stream of Gemma 2 9b. The scorer and the explainer model were Llama 3.1b 70b instruct, quantized to 4 bits.

A.5.4 Dependence on the origin of examples

Sampling the examples from the whole distribution of activations, or sampling a fixed number of examples over different activation quantiles, significantly improved the scores, compared with sampling only from the top examples. These evaluations were done using the 131k latent SAE trained on the residual stream of Gemma 2 9b. The scorer and the explainer model were Llama 3.1b 70b instruct, quantized to 4 bits.

A.5.5 Dependence on the number of examples

The number of examples shown to the model seems to saturate, at least with the explainer model we used. These evaluations were done using the 131k latent SAE trained on the residual stream of Gemma 2 9b. The scorer and the explainer model where Llama 3.1b 70b instruct, quantized to 4bit.

A.5.6 Explainability across layers

We find that, in the case of the 131k latent SAE trained on the residual stream, the earliest layers have lower scores than the later layers. These evaluations were done using the 131k latent SAE trained on the residual stream of Gemma 2 9b. The scorer and the explainer model where Llama 3.1b 70b instruct, quantized to 4bit.

A.5.7 Dependence on size of explainer and scorer models

Having a larger explainer model leads to better explanations, but our results suggest that it tends to saturate. It is possible that that is due to the fact that the prompt and setup was optimized for Llama 3.1 70b, and that better explanation could be extracted using Claude Sonnet 3.5 if the prompt was

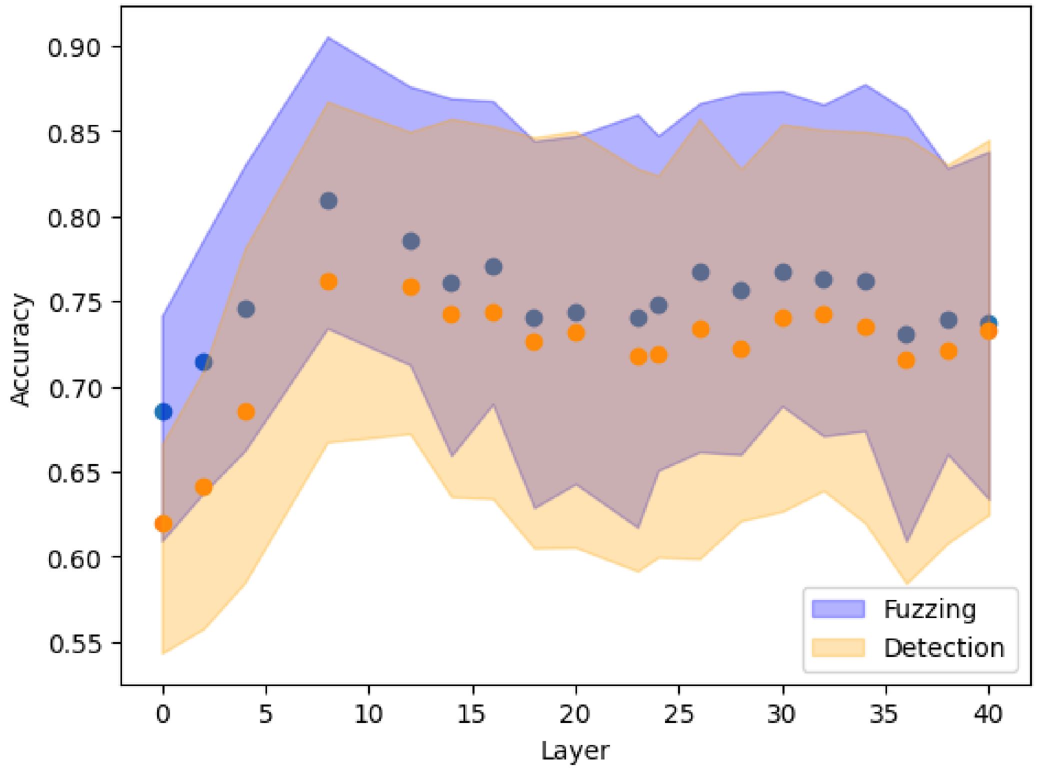


Figure A4: **Accuracy on fuzzing and detection scoring** Dots correspond to the median score over c.a. 300 latent explanations, and colored region denotes the interquartile range.

Table A5: Impact of example sampling strategies on Fuzzing and Detection performance. Numbers shown are the median score and the interquartile (25%-75%) range.

Experiment	Fuzzing	Detection	Embedding
Random explanation	0.51 (0.45–0.57)	0.51 (0.45–0.58)	0.51 (0.44–0.57)
Randomly initialized Topk SAE	0.55 (0.50–0.60)	0.54 (0.50–0.59)	–
Randomly sampled	0.76 (0.68–0.86)	0.74 (0.62–0.84)	0.66 (0.56–0.78)
Sampled from quantiles	**0.77 (0.69–0.87)**	**0.74 (0.64–0.85)**	0.68 (0.57–0.80)
Sampled from top examples	0.73 (0.64–0.83)	0.72 (0.62–0.82)	**0.70 (0.58–0.80)**

Table A6: Impact of number of examples on Fuzzing and Detection performance. Numbers shown are the median score and the interquartile (25%-75%) range.

Experiment	Fuzzing	Detection	Embedding
Random explanation	0.51 (0.45–0.57)	0.51 (0.45–0.58)	0.51 (0.44–0.57)
Randomly initialized Topk SAE	0.55 (0.50–0.60)	0.54 (0.50–0.59)	–
Shown 10 examples	0.73 (0.62–0.85)	0.71 (0.58–0.82)	0.64 (0.54-0.74)
Shown 20 examples	0.74 (0.64–0.85)	0.72 (0.60–0.84)	0.66 (0.54-0.76)
Shown 40 examples	**0.76 (0.67–0.86)**	**0.74 (0.63–0.85)**	**0.68 (0.57–0.80)**
Shown 60 examples	0.75 (0.66–0.85)	0.73 (0.62–0.84)	0.68 (0.57–0.79)

better tuned for it. We find that having a smaller model score significantly drops the scores of the explanations—as bad as having the scorer model generate the explanations and the bigger model score them.

A.5.8 Dependence on SAE size and location

We compare the interpretability of SAEs with different number of latents, trained on the same model, with the neurons of that model. We find that the SAE with the highest number of latents to have the highest scores in the case of the Gemma 2 9b SAEs and that SAEs trained on the residual stream have higher scores for both Gemma 2 9b and Llama 3.1 8b.

The scorer and the explainer model were Llama 3.1 70b instruct, quantized to 4bit.

A.6 Alignment of latents using decoder weights

See figure A5 for a depiction of the alignment between weights of decoders at various layers. Here, the alignment is measured using the Frobenius inner product between the decoder weight matrices. Due to the permutation symmetry of SAEs layers are not aligned by default, but can't be aligned using the Hungarian algorithm. We observe that the decoder weights of SAEs trained on the residual stream can be better aligned than those trained on the MLP.

A.7 Automatically interpreting interventions

A.7.1 Scoring implementation

For each feature, we sample a pool of 40 length-64 prompts from RPJv2 (Computer, 2023), of which 30 i.i.d. prompts are taken for scoring, while the remaining 10 are used by the explainer. The pool is sampled among nonzero activating contexts, stratified by the quintile of the feature's max activation in that context (i.e., the nonzero activating documents are sorted by the context's maximum activation of the current feature, then 8 contexts are sampled from the first quintile, 8 from the second etc.). Each of these prompts is then truncated to include only the first token on which the feature activates, so that all previous activations for the feature are 0. Activations on the first token position are ignored because the SAEs were not trained on that position.

We filter out features that activates on less than 200 of the 10 million RPJ-v2 tokens.

Table A7: Comparison of explainer models for Fuzzing and Detection. Numbers shown are the median score and the interquartile (25%-75%) range.

Experiment	Fuzzing	Detection	Embedding
Random explanation	0.51 (0.45–0.57)	0.51 (0.45–0.58)	0.51 (0.44–0.57)
Randomly initialized Topk SAE	0.55 (0.50–0.60)	0.54 (0.50–0.59)	–
Claude explaining	0.75 (0.68–0.84)	**0.75 (0.65–0.85)**	0.70 (0.58–0.81)
Llama 70b explaining (4 bit)	**0.76 (0.67–0.86)**	0.74 (0.63–0.85)	0.67 (0.57–0.80)
Llama 70b explaining (4 bit) 8b scoring	0.69 (0.61–0.77)	0.69 (0.59–0.79)	–
Llama 8b explaining	0.70 (0.60–0.81)	0.70 (0.59–0.81)	0.64 (0.54–0.75)
Llama 8b explaining, 8b scoring	0.67 (0.59–0.75)	0.67 (0.57–0.77)	–
Human explaining	0.75 (0.66–0.85)	0.74 (0.64–0.85)	**0.71 (0.62–0.81)**

Table A8: Comparison of SAEs for Fuzzing and Detection. Numbers shown are the median score and the interquartile (25%-75%) range.

Experiment	Fuzzing	Detection
Random explanation	0.51 (0.45–0.57)	0.51 (0.45–0.58)
Randomly initialized Topk SAE	0.55 (0.50–0.60)	0.54 (0.50–0.59)
131k latents Gemma 2 9b	**0.76 (0.67–0.86)**	**0.74 (0.63–0.85)**
16k latents Gemma 2 9b	0.73 (0.63–0.83)	0.70 (0.59–0.79)
Top 32 neurons Gemma 2 9b	0.62 (0.54–0.70)	0.59 (0.53–0.64)
Top 256 neurons Gemma 2 9b	0.59 (0.53–0.65)	0.57 (0.52–0.62)
262k latents Llama 3.1 8b MLP	0.79 (0.69–0.86)	0.79 (0.64–0.85)
262k latents Llama 3.1 8b	**0.81 (0.71–0.86)**	**0.83 (0.68–0.85)**
Top 32 neurons Llama 3.1 8b	0.55 (0.51–0.62)	0.53 (0.50–0.59)
Top 256 neurons Llama 3.1 8b	0.54 (0.49–0.60)	0.53 (0.50–0.57)

We then sample generations with a maximum of 8 new tokens from the subject model. For generations with the intervention, we perform an additive intervention on the feature at all token positions after and including the final prompt token. We tune the intervention strength of each feature to various KL-divergence values on the scoring set with a binary search. We stop the binary search when the KL divergence is within 10% of the desired value[4]. For our zero-ablation experiment, instead of doing an additive intervention we clamp the feature's activation to 0. Specifically, the hidden states are encoded by the SAE, then the SAE reconstruction error is computed using a clean decoding, then the SAE encoding is clamped and decoded, and the error is added to the clamped decoding.

The scorer is Llama-3.1 8B base. We use the base model for improved calibration, and prompt it as follows.

```
<PASSAGE>
from west to east, the westmost of the seven wonders of the
world is the great wall of china

The above passage contains an amplified amount of "Asia"

<PASSAGE>
Given 4x is less than 10, 4

The above passage contains an amplified amount of "numbers"

<PASSAGE>
```

[4]Sometimes the KL-divergence is not perfectly monotonic in the intervention strength so 10% error is exceeded. We report the average KL divergence that we observe in figure A6

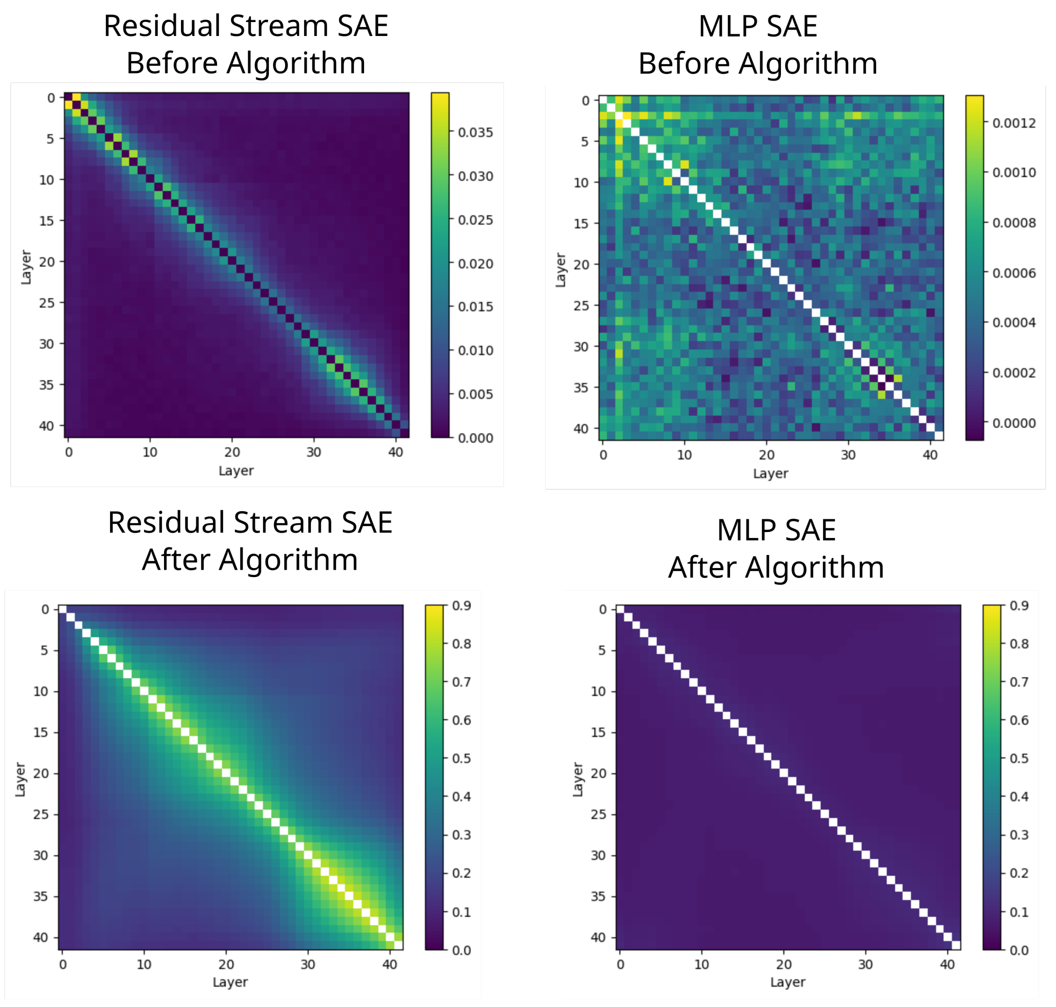


Figure A5: **Alignment between decoder directions of the 16k Gemma 2 9b SAEs** Because each layer is trained independently, there is no correspondence between latents at different layers. It is possible to align them using the Hungarian algorithm, but even still the MLP SAEs are less aligned than the residual stream ones.

```
In information theory, the information content, self-
information, surprisal, or Shannon information is a basic
quantity derived by her when she was a student at Windsor

The above passage contains an amplified amount of
"she/her pronouns"

<PASSAGE>
My favorite food is oranges

The above passage contains an amplified amount of
"fruits and vegetables"

<PASSAGE>
...
```

A.7.2 Generating explanations

We generate explanations using only the intervention's effect on the subject's next-token probabilities because this leads to a concise and precise prompt. The explainer, like the scorer, is Llama-3.1 8B.

The explainer sees a distribution of 10 prompts that is sampled i.i.d. from the same population as the scorer's prompts.

We use the following prompt for the explainer, with 3 few-shot examples truncated for brevity.

```
We're studying neurons in a transformer model. We want to know
how intervening on them affects the model's output.

For each neuron, we'll show you a few prompts where we
intervened on that neuron at the final token position, and the
tokens whose logits increased the most.

The tokens are shown in descending order of their probability
increase, given in parentheses. Your job is to give a short
summary of what outputs the neuron promotes.

Neuron 1
<PROMPT>Given 4x is less than 10,</PROMPT>
Most increased tokens: ' 4' (+0.11), ' 10' (+0.04),
' 40' (+0.02), ' 2' (+0.01)

<PROMPT>For some reason</PROMPT>
Most increased tokens: ' one' (+0.14), ' 1' (+0.01),
' fr' (+0.01)

<PROMPT>insurance does not cover claims for accounts
with</PROMPT>
Most increased tokens: ' one' (+0.1), ' more' (+0.02),
' 10' (+0.01)

Explanation: numbers

Neuron 2
...
```

A.7.3 Baselines

Random SAE. We experiment with a random top-k SAE with $k = 50$, roughly the average sparsity of the gemma SAEs. The encoder is initialized with 131,072 spherically uniform unit-norm features, and the decoder is initialized to its transpose. We use a random SAE with TopK activations because we need sparsity for our sampling procedure to work properly.

Random explanations. For each layer and target KL value, we compute a random explanation baseline where we shuffle the explanations across features.

A.7.4 Intervention interpretability results

Figure A6 shows the average intervention score at each layer, for a few KL values. Intervention scores are higher in later layers, likely because of the increased proximity to the model's output.

Figure A7 is similar to the right half of figure 4, but at multiple layers and KL values. Early layers have small intervention scores across the board, so there is little correlation, while the reason for little correlation in layer 41 is less clear.

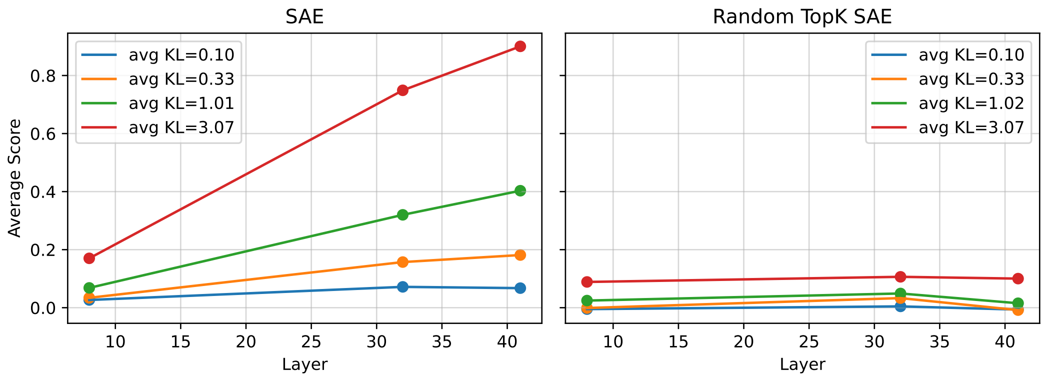


Figure A6: Average intervention score vs layer. SAE features in later layers have more explainable effects on output. Random features, however, have uninterpretable effects on output, even at late layers.

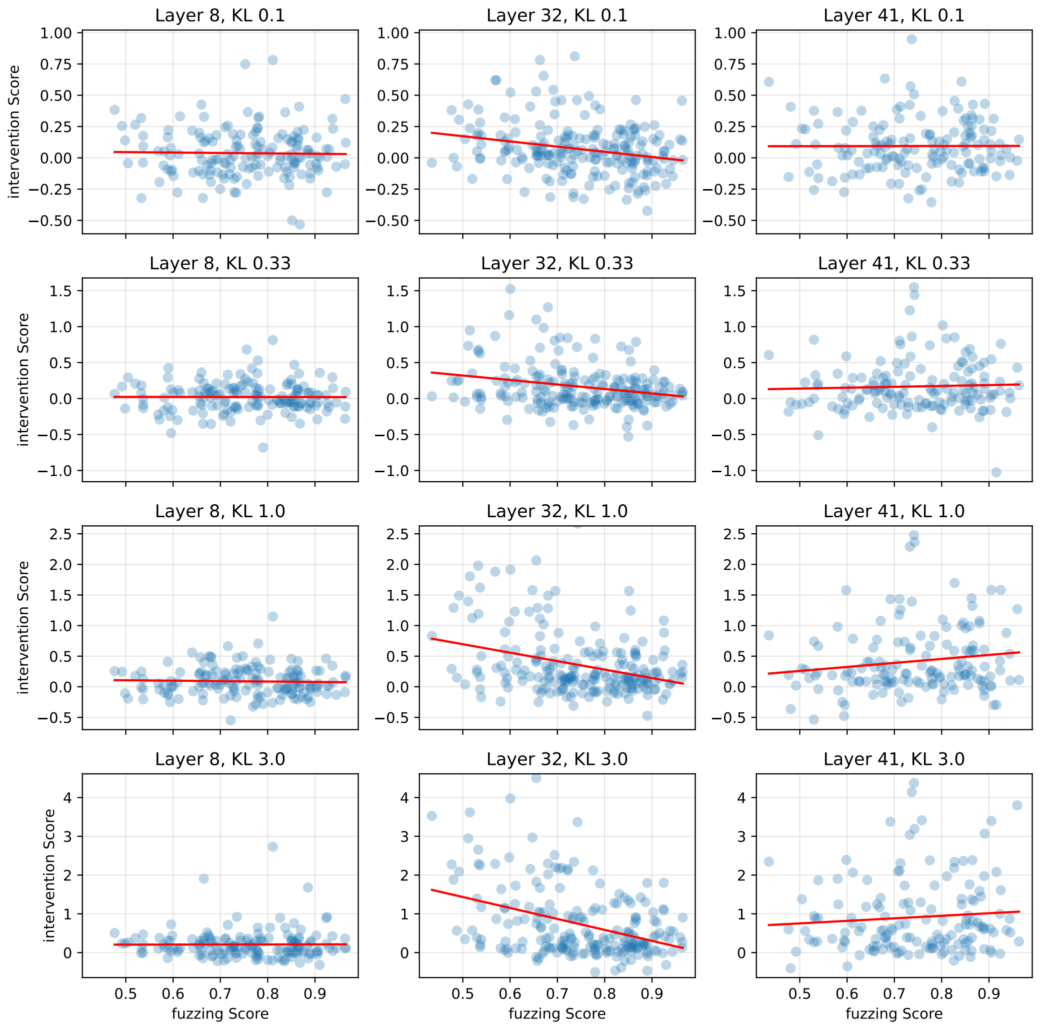


Figure A7: Comparison of intervention and fuzzing scores across layers at various target KL values.