Features that Make a Difference: Leveraging Gradients for Improved Dictionary Learning

Anonymous ACL submission

Abstract

Sparse Autoencoders (SAEs) are a promising approach for extracting neural network representations by learning a sparse and overcomplete decomposition of the network's internal activations. However, SAEs are traditionally trained considering only activation values and not the effect those activations have on downstream computations. This limits the information available to learn features, and biases the autoencoder towards neglecting features which are represented with small activation values but strongly influence model outputs. To address this, we introduce Gradient SAEs (g-SAEs), which modify the k-sparse autoencoder architecture by augmenting the TopK activation function to rely on the gradients of the input activation when selecting the k elements. For a given sparsity level, g-SAEs produce reconstructions that are more faithful to original network performance when propagated through the network. Additionally, we find evidence that g-SAEs learn latents that are on average more effective at steering models in arbitrary contexts. By considering the downstream effects of activations, our approach leverages the dual nature of neural network features as both *representations*, retrospectively, and *actions*, prospectively. While previous methods have approached the problem of feature discovery primarily focused on the former aspect, g-SAEs represent a step towards accounting for the latter as well.

1 Introduction

Sparse Autoencoders (SAEs) have emerged as a promising method for interpreting neural networks, aiming to recover a model's *features* via dictionary learning (Bricken et al., 2023; Cunningham et al., 2023; Templeton et al., 2024). While there is no universally accepted definition of features, they are generally understood to be the atomic units of language models' computation, possessing the quality of monosemanticity as both representations and causal intermediates. The success of SAEs in identifying directions in language model activation space that are causally relevant and interpretable provides evidence that a significant fraction of the internal activations of language models are sparse, linear combinations of vectors which are each oriented in a direction corresponding to a feature of the model. (Park et al., 2024b,a). The elements of the SAE-learned dictionary corresponding to these directions we call *latents*.

However, SAEs are trained to encode input activations such that the reconstruction loss over many tokens from a training corpus is minimized. This has raised some concerns that SAEs may not learn latents corresponding to the features of a model, but rather the features of a dataset (Braun et al., 2024; Dooms and Wilhelm, 2024). That is, SAEs might give the illusion of interpretability by learning latents based on frequent and distinct concepts in the *training corpus*, rather than learning latents because they correspond to features that play an important and distinct role in the model's decision-making process. We would like a better guarantee that our interpretability tools discover the features of a model, rather than primarily reflecting the training data.

Additionally, if one is interested in interpretability as a means to exert more fine-grained control over the behavior of models, it would be desirable to have interpretability tools that are biased towards uncovering the features that are most responsible for a model's output. Templeton et al. (2024) speculate that current Large Language Models (LLMs) may represent orders of magnitude more features than the size of the dictionaries of the largest SAEs trained to date. The limited capacity of SAEs incentivizes them to neglect features that strongly affect model outputs if overlooking them usually results in small reconstruction errors (Braun et al., 2024).

These desiderata motivate our approach, which

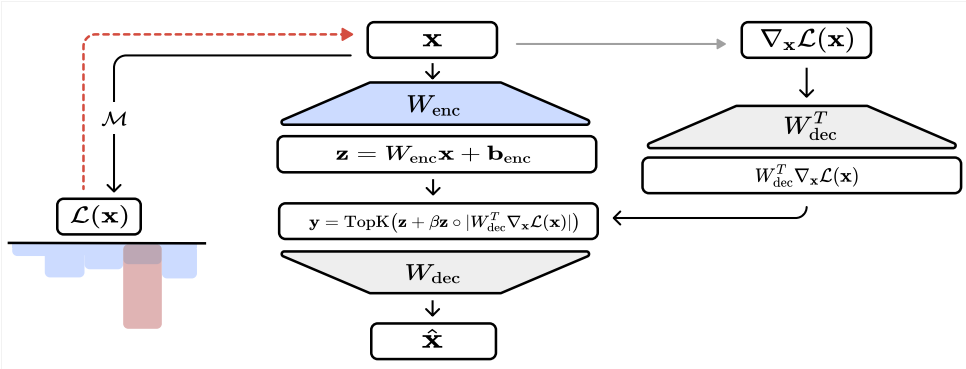

Figure 1: The setup of g-SAE training. Here $\mathcal{L}(\mathbf{x})$ is the function mapping the residual stream activation at the layer the SAE was trained on, to the predictive cross entropy loss it yields. The red dotted line denotes backpropagation.

aims to address the challenges above by connecting dictionary learning more closely with model *outputs*. In this paper we show:

1. **Distance in LLM activation space correlates poorly with degree of downstream effect, but gradients can yield accurate first order predictions.** We show that there is only a weak correlation between the norm of a perturbation vector and it's effect on LLM downstream outputs, as measured by change in cross-entropy loss. In contrast to the norm alone, the first order approximation given by gradients of activations with respect to loss strongly predict downstream effects for perturbations of moderate size. This suggests that reconstruction loss minimization doesn't directly incentivize the SAE to learn reconstructions that minimize difference in effect.

2. **In SAE training, incorporating gradients produces improvements on various Pareto frontiers defined by popular SAE architectures, and results in more complete use of the SAE's capacity.** For a given expansion size, using a gradient-aware TopK activation function leads to improvements in the minimization of difference in model loss when reconstructed outputs are propagated downstream. It also leads to fewer permanently inactive units of the SAE hidden layer.

3. **g-SAE latents provide superior steering with no cost to interpretability.** g-SAEs recover latents that appear to be as interpretable as existing architectures, and on average are more influential to the specific logits they are associated with, when used as steering vectors in arbitrary contexts.

2 Motivation and Background

2.1 SAE Preliminaries

The core of an SAE is the following: an encoder matrix which projects the input activation into a higher dimensional latent space, where the elements represent a dictionary of learned features, and a decoder which reconstructs the input from this sparse vector. Formally, for a model's activation vector $\mathbf{x} \in \mathbb{R}^d$,

$$\mathbf{y} = \text{ReLU}(W_{\text{enc}}(\mathbf{x} - \mathbf{b}_{\text{dec}}) + \mathbf{b}_{\text{enc}})$$
$$\hat{\mathbf{x}} = W_{\text{dec}}\mathbf{y} + \mathbf{b}_{\text{dec}}$$

where $W_{\text{enc}}\mathbf{x} \in \mathbb{R}^h$, h is the size of the dictionary, and $W_{\text{enc}} \in \mathbb{R}^{h \times d}$. We will denote the sparse latent vector $W_{\text{enc}}\mathbf{x}$ as $\mathbf{y}$ and the reconstructed activation as $\hat{\mathbf{x}}$. The elements of $\mathbf{y}$ are the SAE's latents.

Generally, SAEs are trained with a loss function defined by the minimization of the reconstruction error $\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2$ and some mechanism for promoting sparsity. To this end, Bricken et al. (2023) use a term in the loss function to minimize the L_1 norm of $\mathbf{y}$. This approach comes with some downsides, including the phenomenon of *shrinkage*, that is, the unintended optimization of the elements of $\mathbf{y}$ to be minimized in addition to promoting the sparsity of $\mathbf{y}$. Various alternative architectures have been proposed, which we detail briefly below.

2.2 Relevant Work

Gao et al. (2024) introduced the TopK autoencoders architecture which use a TopK activation function to serve the purpose of direct sparsity enforcement. This activation function zeroes out all but the k most active elements of the pre-activation $\mathbf{z}$. Various other architectures have been proposed in addition to the TopK architecture to improve

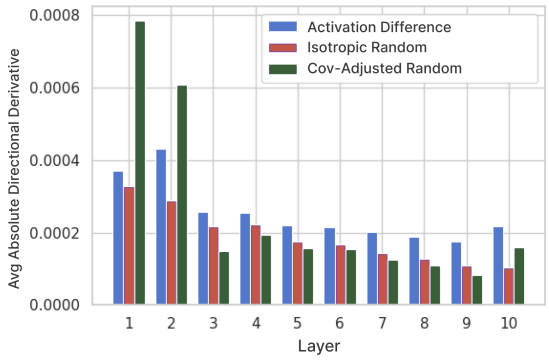

Figure 2: Comparison of directional derivatives in GPT-2's residual stream. The plot shows the absolute value of the directional derivatives towards activation differences vs. random directions. Layer n corresponds to the n_{th} `resid_post` hookpoint. Feature directions exhibit consistently higher derivatives with respect to loss, particularly in later layers.

upon the basic sparse autoencoder setup, including Gated SAEs, JumpReLU SAEs, Tokenized SAEs and Switch SAEs Rajamanoharan et al. (2024a,b); Dooms and Wilhelm (2024); Mudide et al. (2024), (Rajamanoharan et al., 2024b). Recent research has also explored the challenge of identifying functionally significant features. In particular, Braun et al. (2024) posited that reconstruction loss may not strongly incentivise an SAE to learn latents that explain a model's performance. To address this limitation, they developed end-to-end (e2e) SAEs. Rather than using MSE as a loss function, e2e SAEs optimize for explanation of downstream effect by minimizing the KL divergence between the output logits of the original activations and those of the SAE-reconstructed activations. However, e2e SAE latents appear to be less influential compared to those derived from standard SAE architectures. Lee and Heimersheim (2024) hypothesize that this is because e2e SAEs optimize their objective by exploiting the space outside the typical activation space.

2.3 LLM outputs are especially sensitive in the direction of their features

The linear representation hypothesis (Park et al., 2024b,a; Elhage et al., 2022) posits that deep learning models represent features as linear directions in activation space. If this is the case, and here we will assume it is, then it also follows that the difference between two real activation vectors corresponds to a linear combination of feature directions. Previous work has analyzed the sensitivity of GPT-2's (Radford et al., 2019) output to perturbations in random directions vs. perturbations in the direction of the difference between two real activation vectors (Heimersheim and Mendel, 2024; Giglemiani et al., 2024). This work suggests that LLM outputs are significantly more sensitive to finite perturbations in the direction of linear combinations of features than perturbations in random directions.

We tested whether this heightened sensitivity to linear combinations of feature directions implies that the directional derivative of activations in these directions is particularly large as well; for arbitrary residual stream activations $\mathbf{x}$ of GPT-2 within a sequence, we compute $\nabla_{\mathbf{x}}\mathcal{L}(\mathbf{x})$ where $\mathcal{L}$ is the function mapping $\mathbf{x}$ to the corresponding cross entropy loss it yields. We then compare the average directional derivative in random directions and in the direction of arbitrary activation differences $\mathbf{x}_i - \mathbf{x}_j$. We use two different kinds of random directions; isotropically random directions[1], and covariance-adjusted random directions that mirror the covariance structure of real activations (Lee and Heimersheim, 2024).

For layers 3 onward, where we expect sophisticated features to exist, the activation difference directions have significantly higher gradient products than both isotropic random and covariance-adjusted random directions, illustrated in Figure 2. This confirms previous findings that the local linear approximation of output change provided by gradients are accurate on the relevant scales (Syed et al., 2023; Templeton et al., 2024). This consistent pattern for later layers motivates a key hypothesis behind our method:

A model's output is particularly locally sensitive towards feature directions.

2.4 First order approximations are a strong predictor of effect

Standard SAEs use activation values to learn their decompositions of model activations. Are the magnitudes of activation values a reliable guide to influence on outputs?

We find that in all MLP output layers of the LLMs we tested, there is generally a weak correlation between the norm of an isotropic random permutation to an MLP output, and the resulting change to prediction cross-entropy loss (Fig. 3). In the context of SAEs, this low correlation means that

[1] Here, isotropic means that the direction of the permutation is uniformly random across components in the standard basis, such that no component is favored.

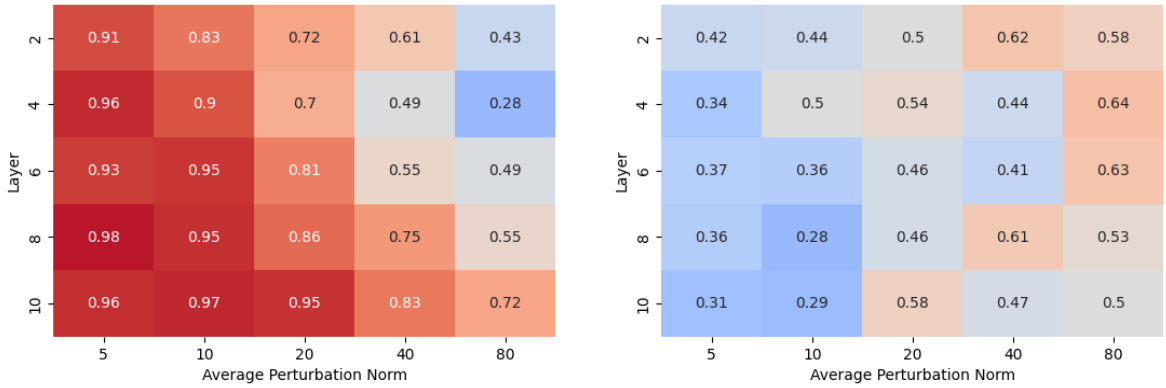

Figure 3: **Left:** Spearman correlation coefficients between change in cross-entropy loss $\Delta\mathcal{L}(\mathbf{x})$ from isotropically perturbing an MLP output, and the first order approximation of that change ($|\nabla_{\mathbf{x}}\mathcal{L}(\mathbf{x}) \cdot \delta\mathbf{x}|$). **Right:** Correlations between the norm of a perturbation and the resulting effect on loss. Perturbations for each column are drawn from a uniform distribution with the displayed mean and a standard deviation of one-half the mean. Data from `resid_post` hookpoint of GPT-2 across diverse tokens.

"not all reconstruction errors are equal." Frequently, small perturbations will cause large downstream effects, and conversely, large perturbations can be ineffectual. Indeed, we find that the effect of a given perturbation can vary by as much as 3 orders of magnitude depending on its direction. The standard SAE training setup makes no distinction between reconstruction errors if they have the same norm. Additionally, conditional upon model outputs being especially sensitive in feature directions, the standard SAE training setup leaves valuable information on the table in the search for features.

On the other hand, the first order approximation of the perturbation which leverages the gradient has, in general, a stronger correlation with the resulting effect, especially at later layers and smaller perturbations. Given that SAEs generally produce reconstruction errors with norms on the lower end of this range, we hypothesize that in the context of SAE training, gradients carry reliable information about the relative influence on model output of different directions in activation space, which we can leverage for better dictionary learning.

3 g-SAEs

A g-SAE is a standard SAE but with the following activation function:

$$\sigma(\mathbf{z})_i = \begin{cases} \mathbf{z}_i & \text{if } i \in K \\ 0 & \text{otherwise} \end{cases}$$

$$K = \text{TopK}\left(\mathbf{z} + \beta\mathbf{z} \circ \left|W_{\text{dec}}^T \cdot \nabla_{\mathbf{x}}\mathcal{L}(\mathbf{x})\right|\right)$$

where $\mathcal{L}(\mathbf{x})$ is the function mapping a residual stream activation to its associated predictive cross entropy loss, W_d is the decoder matrix, and β is a tunable hyperparameter. This activation function selects k latents $\mathbf{z}_i$ based on two criteria: 1) the value of $\mathbf{z}_i$, and 2) $\mathbf{z}_i$ times the i_{th} component of $W_d^T \cdot \nabla_{\mathbf{x}}\mathcal{L}(\mathbf{x})$, which is the linear transformation of the gradient at $\mathbf{x}$ into the SAE latent space. The latter term can be thought of as the attribution of $\mathbf{z}$ with respect to the model loss; it resembles the attribution vector derived by attribution patching (Nanda, 2024)[2]. The term $\mathbf{z} \circ \left|W_d^T \cdot \nabla_{\mathbf{x}}\mathcal{L}(\mathbf{x})\right|$ represents the local linear approximation of the effect that turning latent $\mathbf{z}_i$ off has on the loss yielded by $\mathbf{x}$.

These dual criteria place productive constraints on both the encoder and decoder: when β is high, the decoder is forced to work with only latents corresponding to directions in the input space with large effects as estimated by the local linear approximation, and therefore learn to reconstruct $\mathbf{x}$ by leveraging these directions. Likewise, with high β, the encoder is only updated along rows corresponding to high-influence latents, (because in backpropagation, no gradients flow through the zeroed-out latents) and consequently, it must learn a decomposition of $\mathbf{x}$ along directions with high attributions with respect to the model's loss. Thus, while the SAE loss remains reconstruction error, the SAE is biased towards learning latents oriented towards high-influence directions. A major advantage of

[2]Key differences between attribution here and attribution in Nanda (2024) are that we effectively use a baseline value of 0 for the feature instead of a baseline value taken from the feature's activity on a second prompt. Additionally, gradients of $\mathbf{z}$ are never taken, rather the gradient of x is projected into the SAE latent space with the decoder

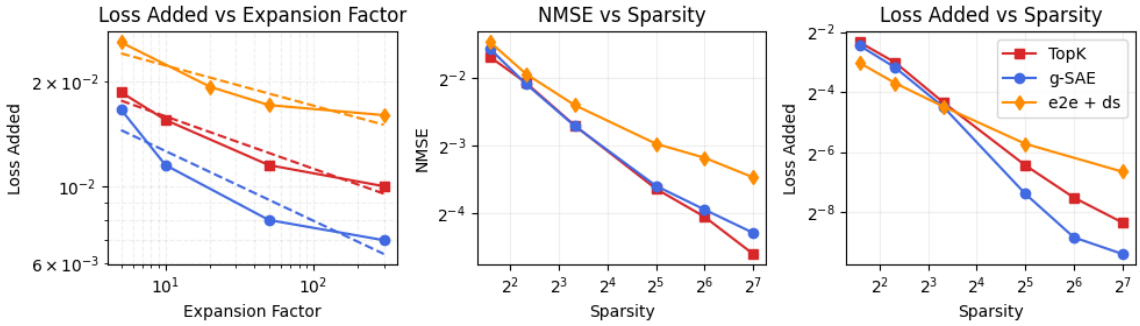

Figure 4: **Left:** Number of latents vs Loss Added for L_0=32. **Middle:** L_0 against NMSE holding the number of latents fixed to 15360. **Right:** L_0 against Loss Added, holding number of latents fixed. All SAEs were trained on GPT-2 small with ∼14M tokens.

our method is that this is accomplished without incorporating gradients with respect to $\mathbf{x}$ into the loss function of the SAE, which we find tends to be unstable and expensive.

The experiments in this section are all performed on layer 7 of GPT-2 small. Throughout training a variety of machines were used including one with 8 A100 GPUs training for two to three hours per model. For g-SAEs, we use $\beta = 10^5$. Following Dooms and Wilhelm (2024), we use the added cross-entropy loss to measure the impact on the model prediction and normalized MSE to measure reconstruction:

$$\mathcal{L}_{\text{added}}(\mathbf{x}) = \frac{\mathcal{L}(\hat{\mathbf{x}}) - \mathcal{L}(\mathbf{x})}{\mathcal{L}(\mathbf{x})}, \text{NMSE}(\mathbf{x}) = \frac{\|\mathbf{x} - \hat{\mathbf{x}}\|_2}{\|\mathbf{x}\|_2}$$

4 Results

4.1 Capacity

g-SAEs perform comparably on the NMSE-sparsity frontier, and add slightly less loss for a given sparsity, particularly at denser sparsities. When sparsity is held constant and expansion size varies, g-SAEs display a steeper curve than TopK, indicating increased efficiency in utilizing additional capacity. This suggests that g-SAEs leverage additional latents more productively than pure TopK SAEs.

Related to this is the fact that g-SAEs have fewer inactive, or "dead" latents than other architectures, for a given sparsity. Because of the vacillatory nature of the gradients of hidden layer activations, the TopK$_G$ activation function introduces a natural kind of stochasticity which allows for previously unused latents to receive updates; this may be a mechanism behind the fewer dead latents associated with g-SAEs.

We find that when β is too large, performance degrades in terms of NMSE and loss added, suggesting that both terms in the activation function σ are necessary.

Sparsity	g-SAE	TopK	E2E
3	0.95	0.98	0.95
5	0.88	0.92	0.87
10	0.71	0.81	0.68
32	0.34	0.43	0.29
64	0.19	0.26	0.16

Table 1: A comparison of dead latents against L_0 sparsity. Here a latent is considered dead if it doesn't activate for 5 consecutive batches.

4.2 Steering

A core motivation behind g-SAEs is the desire to learn latents which have an interpretable and strong causal connection with the model's actions. Such a connection would be useful in providing more control over model behavior through the use of latent directions as steering vectors (Li et al., 2024; Turner et al., 2024; Marks and Tegmark, 2024; Panickssery et al., 2024; Conmy and Nanda, 2024; Templeton et al., 2024). Additionally, evidence of a causal relationship between latent activations and model outputs would suggest that latents are more closely tracking model features, if we assume that features have a dual nature as both representations and actions.

To investigate this, for a given alive latent $\mathbf{y}_i$, we collected the set L of the n logits that $\mathbf{y}_i$ is most associated with: L is computed by projecting $\mathbf{y}_i$'s direction into the vocabulary space via the unembedding matrix W_U, yielding a logit vector $W_U^\top \mathbf{y}_i$, and then selecting the n logits corresponding to the largest values in $W_U^\top \mathbf{y}_i$, which represent the vocabulary tokens that $\mathbf{y}_i$ points towards most. Then, we

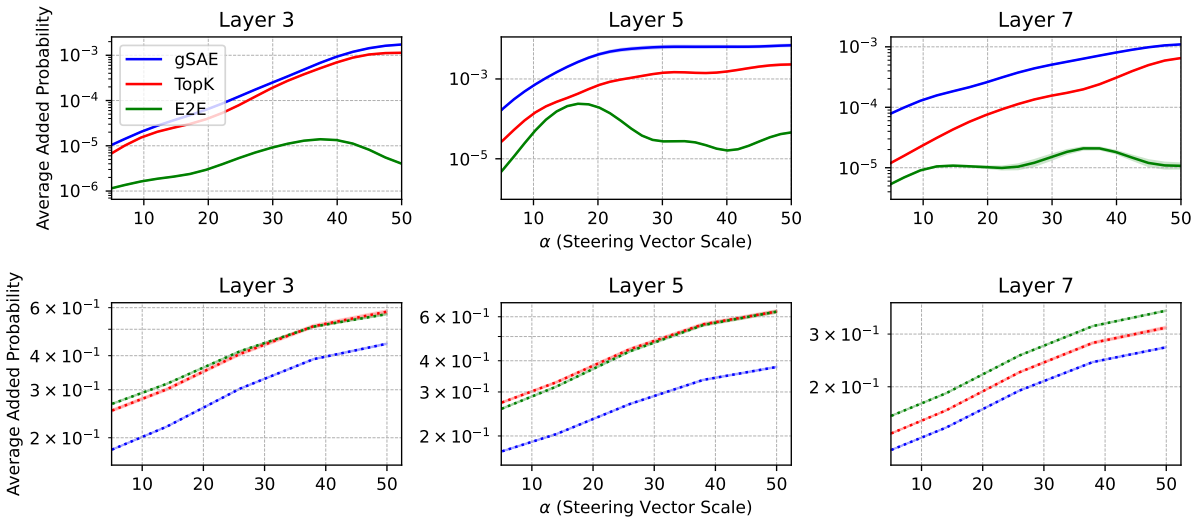

Figure 5: **Top:** The average effect of applying a steering vector in the direction of a latent $\mathbf{y}_i$ on the logits $\mathbf{y}_i$ activates with. *Higher is better.* **Bottom:** The average total probability added to all other logits when applying a steering vector in the direction of $\mathbf{y}_i$. *Lower is better.* Data from a random samples of latents from SAEs with $L_0 = 32$, with α incremented in steps of 5/3 above, and 10 below. Gaussian smoothing applied for better visibility.

compute the following:

$$\sum_{j \in L} \left(\mathcal{M}\left(\mathbf{x} + \alpha \frac{W^i_{\text{dec}}}{\|W^i_{\text{dec}}\|} \right)_j - \mathcal{M}(\mathbf{x})_j \right)$$

Where:
- L is the set of associated logits for latent $\mathbf{y}_i$,
- $\mathcal{M}$ is the function mapping the current layer to the output probability vector
- $\mathbf{x}$ is the original activation,
- α is the scaling factor,
- W^i_{dec} is the i_{th} decoder column

Conceptually, this equation represents the degree of feature-specific influence a steering vector exerts over the output. We see that on average, g-SAE latents more strongly cause increases to their corresponding logits when used as steering directions. This increase is substantial: the added probability of a corresponding logit is on average over 5x greater in g-SAEs than in TopK SAEs (see Fig. 5). Additionally, we tested the specificity of the effect of adding the latent-derived steering vectors by measuring the added probability to all *except* the n most associated logits. Here we see that g-SAEs again display superior performance; they increase the likelihood of arbitrary logits less than other SAEs' steering vectors.

Focusing on just $\alpha = 50$ for layer 7, we plot the kernel density estimation plots for the added probability to the associated logits for 1000 latents from each SAE, over 10 arbitrary input sequences each.

We see in the top plot of Fig. 6 that the right tails of the TopK and g-SAE distributions look quite similar, and that the difference lies in the fact that g-SAEs have fewer latents that fail to affect their associated logits, shifting the mean of the distribution to the right. This suggests that g-SAEs learn more latents which are on the *natural* upper end of the influence distribution, rather than learning latents which have *unnaturally* high influence on outputs (which could be seen as falling prey to Goodhart's law (Strathern, 1997)). Thus, taken together, these experiments offer evidence that g-SAEs learn more latents that are highly influential to their associated set of logits in arbitrary contexts.

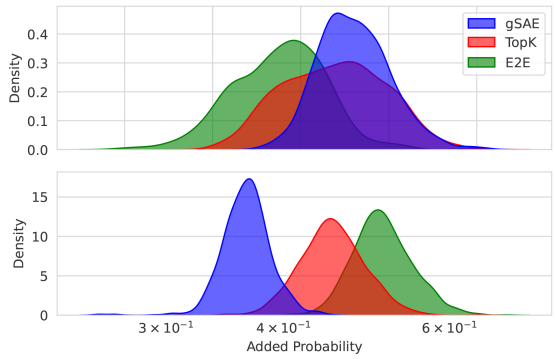

Figure 6: KDE plots of the added probability to associated logits for each SAE type due to the addition of steering vectors, where density is density of latents (Layer 7) **Top:** total probability added to associated logits. **Bottom:** total probability added to all but associated logits

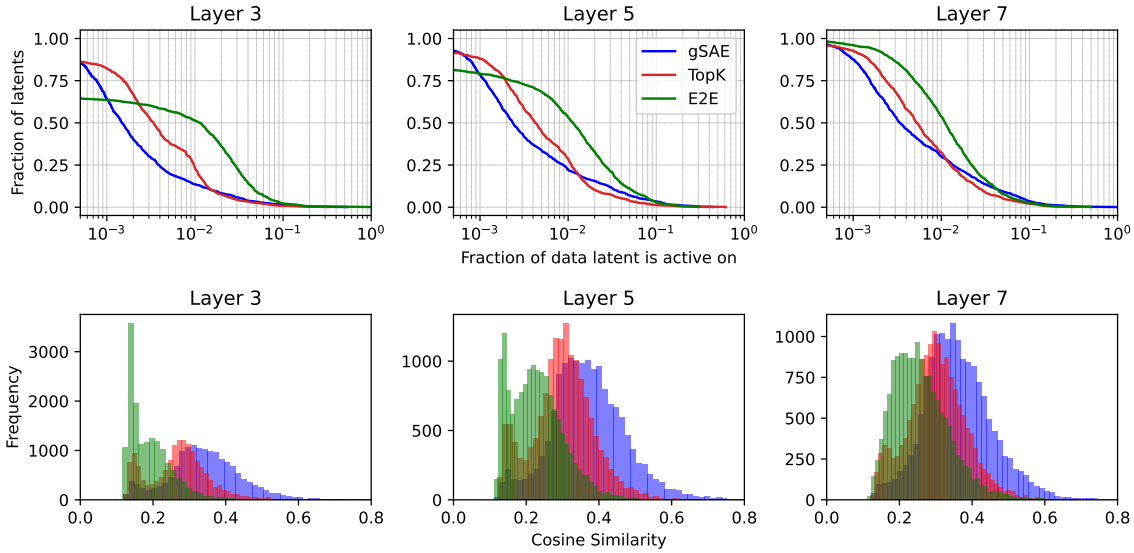

Figure 7: **Top:** Cumulative distribution functions of latent activity for the various SAEs at $L_0 = 32$ residual stream. **Bottom:** The cosine similarity of the most similar decoder column for every column in the SAE decoders. Note that the spikes at the left tail of the histograms likely correspond to dead latents.

4.3 Activity Density Analysis

Intuitively, latents shouldn't be too specific, nor too general. A sufficiently large SAE that learns a feature for every activation could achieve a very low reconstruction loss, however, the latents would be uninformative and likely not reflect the model's representational structure. On the other hand, latents that activate for many (possibly unrelated) patterns of tokens are likewise uninformative. We analyze the activation density of a sample of latents from each type of SAE, which is a measure of how general or specific the latents of an SAE are. Holding the sparsity of the SAEs constant at $L_0 = 32$, the cumulative distribution functions of latent activity (Fig.7) show that g-SAEs tend to have fewer moderately high frequency latents which activate on more than 1% percent of tokens, but more very high frequency latents which activate on more than 10% of tokens. We also see that the median latent of g-SAEs is more specific than than the others, with a median activation frequency of 0.24%, compared to 0.43% and 1.10% for t and e2e SAEs, respectively.

4.4 Feature Splitting

Bricken et al. (2023) and Cunningham et al. (2023) identified the phenomenon of *feature splitting*, in which an SAE learns several latents which are associated with inputs that have little to no apparent semantic difference between them. Braun et al. (2024) suggest that the cosine similarities between each SAE dictionary latent and next-closest latent in the same dictionary is a reasonable proxy for feature splitting, since latents which split features tend to be oriented in highly similar directions.

In Fig. 7 we see that the decoder column similarity histogram of g-SAEs has a higher mean than other SAE types. To investigate this phenomenon further, we take a sample of latents from the left, middle, and right tails of the top decoder similarity histograms and calculate the average directional derivative in the direction of these decoder columns with respect to $\mathcal{L}(\mathbf{x})$ over many activations $\mathbf{x}$. The results in Table 2 suggest that in g-SAEs (as well as e2e SAEs), feature directions are more tightly clustered in high-influence directions. This might indicate that the SAE is allocating more latents towards these directions, such that the density of latents in these high influence directions is higher. Whether this phenomenon leads to more problematic instances of feature splitting in g-SAEs is unclear, and could be the subject of further research.

4.5 Manual Interpretability

To ensure that the increased functional influence of g-SAE latents don't come at the cost of their monosemanticity as representations, we follow a double blind experimental setup similar to that of Rajamanoharan et al. (2024a) and Rajamanoharan et al. (2024b) to measure the interpretability of g-SAE latents according to human raters. We selected three SAE architectures (TopK, E2E TopK,

SAE Type	Left Tail	Middle	Right Tail
g-SAE	2.070	2.538	2.861
TopK	2.187	2.313	2.366
E2E	2.221	2.502	3.428

Table 2: Average absolute directional derivatives ($\times 10^{-5}$) by position in latent similarity histogram for different SAE Types (Layer 7). Here we take a sample of 30 latents for each category and use 300 token sequences to compute the averages.

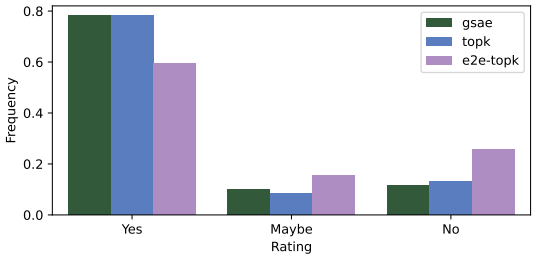

Figure 8: Human evaluation scores for feature interpretability.

g-SAE), trained on the residual stream of GPT-2 small at multiple layers (3, 5, 7, 9), for a total of 16 SAEs. We held TopK sparsity at 32, and our input expansion factor at 20.

Our 3 human raters were presented with samples of activating examples in a random order, with summary information for each feature, and tasked with deciding whether each feature is monosemantic, with answers being 'Yes', 'Maybe' and 'No'. We collected 384 samples, or 32 per SAE.

Results We present our results in Fig. 8. Our reviewers rated g-SAEs and TopK SAEs as similarly interpretable, while e2e-TopK SAEs were slightly less so.

5 Conclusion

Our experiments show that introducing gradient information into the activation function of SAEs produces improvements in output explained at a given expansion size and sparsity. Additionally, the resulting SAE latents appear to have an increased ability to steer model outputs in a concentrated manner in arbitrary contexts by consistently increasing the probability of the associated set of tokens. We believe this improvement is significant both for practical applications aiming to exert more fine-grained control over model outputs, and for interpretability applications that aim to uncover the primary units of models' representations and actions. We find that this improvement comes at no measurable cost to semantic interpretability to human reviewers.

We believe that these developments will lead to dictionary learning that more fully accounts for both the representational and causal aspects of models' features, ultimately contributing to a better understanding of the fundamental workings of LLMs.

6 Limitations

Here we discuss limitations of g-SAEs, along with a discussion of the ethical considerations.

- **Activation Plateaus:** There is evidence that models may be robust to small perturbations around activations, as a consequence of emergent robustness against small noise (Giglemiani et al., 2024). Despite our evidence that gradients in feature directions are higher than those of baseline directions, it may be that such regions of noise-robustness are locally flat enough that the local gradient doesn't provide information about the sensitivity of the model to larger perturbations in the vicinity. If this is the case, we would plausibly want to add a term to the activation function that privileges directions with a large higher order derivatives, but it isn't clear how to do so efficiently.

- **Computational Complexity:** One of the potential drawbacks of g-SAEs is their theoretical computational cost, as both training and running inference on them requires a partial forward pass and backpropagation.

  In practice, we found that this was not a computational bottleneck and there was little change in cost compared to TopK SAEs when training on GPT-2 small.

- **Increasing Scale:** While our results are promising for GPT-2 small, further work needs to be done to confirm that the trends we observe hold for larger LLMs and larger SAEs.

- **Evaluations:** There is a significant need within the community for more thorough evaluations that truly test the quality of SAEs and

their potential effect on model behavior. Recent work is making progress in this direction. Karvonen et al. (2024) propose new quantifiable measures for monosemanticity and feature extraction quality using a toy model. Makelov (2024) demonstrate the use of SAEs for model control in the Indirect Object Identification (IOI) task, providing a task to test the quality of SAEs on. While these studies provide valuable insights, further research is needed to develop comprehensive evaluation frameworks that can be applied more generally to SAEs and their quality. In the context of g-SAEs, it remains to be studied how much increased steerability on the *token* level translates to increased steerability on higher levels of abstraction.

6.1 Ethics

The ethical considerations of this paper reflect the broader ethical considerations around LLMs. We use publicly available datasets and models which can represent potentially problematic biases. However, our work does not contribute in any specific way to the propagation of potentially negative ideas present in the data.

A potential negative use case of g-SAEs, as with other sparse autoencoders, is to steer advanced models towards producing harmful outputs. However, we believe that while progress in interpretability likely inherently comes with the possibility of more fine grained control over AI systems for both good and ill, the upside to a deeper understanding of deep learning systems is currently net positive.

7 Acknowledgements

We thank Thomas Dooms and Daniel Wilhelm for their smol-sae library which we modified and used for the training of our SAEs. We also thank Callum McDougal for the sae_vis library which was used in our manual interpretability experiments.

References

Dan Braun, Jordan Taylor, Nicholas Goldowsky-Dill, and Lee Sharkey. 2024. Identifying Functionally Important Features with End-to-End Sparse Dictionary Learning. *arXiv preprint*. ArXiv:2405.12241 [cs].

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. 2023. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*.

Arthur Conmy and Neel Nanda. 2024. Activation steering with saes.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. 2023. Sparse Autoencoders Find Highly Interpretable Features in Language Models. *arXiv preprint*. ArXiv:2309.08600 [cs].

Thomas Dooms and Daniel Wilhelm. 2024. Tokenized SAEs: Disentangling SAE Reconstructions.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. 2022. Toy models of superposition. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2022/toy$_model/index.html$.

Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. 2024. Scaling and evaluating sparse autoencoders. *arXiv preprint*. ArXiv:2406.04093 [cs].

Giorgi Giglemiani, Nora Petrova, Chatrik Singh Mangat, Jett Janiak, and Stefan Heimersheim. 2024. Evaluating synthetic activations composed of sae latents in gpt-2. *Preprint*, arXiv:2409.15019.

Stefan Heimersheim and Jake Mendel. 2024. Activation plateaus sensitive directions in gpt2.

Adam Karvonen, Benjamin Wright, Can Rager, Rico Angell, Jannik Brinkmann, Logan Riggs Smith, Claudio Mayrink Verdun, David Bau, and Samuel Marks. 2024. Measuring Progress in Dictionary Learning for Language Model Interpretability with Board Game Models.

Daniel Lee and Stefan Heimersheim. 2024. Investigating sensitive directions in gpt-2: An improved baseline and comparative analysis of saes.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2024. Inference-time intervention: Eliciting truthful answers from a language model. *Preprint*, arXiv:2306.03341.

Aleksandar Makelov. 2024. Sparse Autoencoders Match Supervised Features for Model Steering on the IOI Task.

Samuel Marks and Max Tegmark. 2024. The geometry of truth: Emergent linear structure in large language model representations of true/false datasets. *Preprint*, arXiv:2310.06824.

Anish Mudide, Joshua Engels, Eric J. Michaud, Max Tegmark, and Christian Schroeder de Witt. 2024. Efficient dictionary learning with switch sparse autoencoders. *Preprint*, arXiv:2410.08201.

Neel Nanda. 2024. Attribution patching: Activation patching at industrial scale.

Nina Panickssery, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Matt Turner. 2024. Steering llama 2 via contrastive activation addition. *Preprint*, arXiv:2312.06681.

Kiho Park, Yo Joong Choe, Yibo Jiang, and Victor Veitch. 2024a. The Geometry of Categorical and Hierarchical Concepts in Large Language Models. *arXiv preprint*. ArXiv:2406.01506 [cs, stat].

Kiho Park, Yo Joong Choe, and Victor Veitch. 2024b. The linear representation hypothesis and the geometry of large language models. *Preprint*, arXiv:2311.03658.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Tom Lieberum, Vikrant Varma, János Kramár, Rohin Shah, and Neel Nanda. 2024a. Improving Dictionary Learning with Gated Sparse Autoencoders. *arXiv preprint*. ArXiv:2404.16014 [cs].

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

Marilyn Strathern. 1997. 'improving ratings': audit in the british university system. *European Review*, 5(3):305–321.

Aaquib Syed, Can Rager, and Arthur Conmy. 2023. Attribution patching outperforms automated circuit discovery. *Preprint*, arXiv:2310.10348.

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. 2024. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. 2024. Activation addition: Steering language models without optimization. *Preprint*, arXiv:2308.10248.