SAEs Are Good for Steering – If You Select the Right Features

Dana Arad[1*] **Aaron Mueller**[2] **Yonatan Belinkov**[1]
[1]Technion – Israel Institute of Technology [2]Boston University
danaarad@campus.technion.ac.il amueller@bu.edu
belinkov@technion.ac.il

Abstract

Sparse Autoencoders (SAEs) have been proposed as an unsupervised approach to learn a decomposition of a model's latent space. This enables useful applications such as steering—influencing the output of a model towards a desired concept–without requiring labeled data. Current methods identify SAE features to steer by analyzing the input tokens that activate them. However, recent work has highlighted that activations alone do not fully describe the effect of a feature on the model's output. In this work, we draw a distinction between two types of features: *input features*, which mainly capture patterns in the model's input, and *output features*, which have a human-understandable effect on the model's output. We propose input and output scores to characterize and locate these types of features, and show that high values for both scores rarely co-occur in the same features. These findings have practical implications: after filtering out features with low output scores, we obtain 2–3x improvements when steering with SAEs, making them competitive with supervised methods.[1]

1 Introduction

Sparse autoencoders (SAEs) have shown promise in extracting human-interpretable features from the hidden states of language models (LMs) (Bricken et al., 2023; Templeton et al., 2024). One appealing usage of SAEs is to enable fine-grained interventions such as generation steering (O'Brien et al., 2024; Durmus et al., 2024; Marks et al., 2025). However, selecting the right features for intervention is an open problem. Current approaches typically select features to steer based on their activation patterns, i.e., the input texts that most strongly activate a given feature (Huben et al., 2024). While

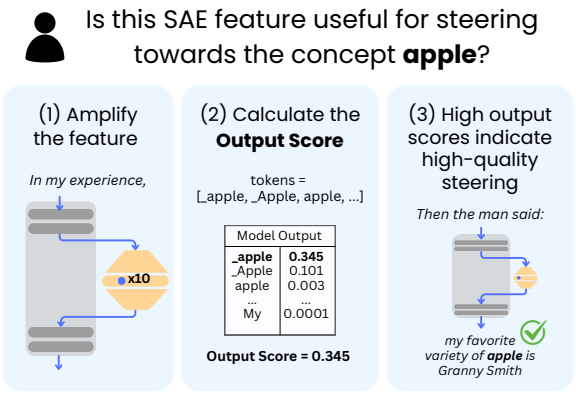


Figure 1: **Selecting features for steering.** (1) Given a concept to steer ("apple"), we amplify a candidate SAE feature during a single forward pass of the model on a neutral prompt. (2) We compute the feature's **output score** based on the rank and probability of representative after intervention. (3) Features with high output scores are more likely to be effective for steering.

input-based activations can reveal meaningful patterns, recent work highlights a critical limitation: a feature's activations are not necessarily the same as its causal effect on the model's output (Durmus et al., 2024; Paulo et al., 2024; Gur-Arieh et al., 2025). As a result, the way features are selected can lead to suboptimal steering, reducing its consistency and reliability (Durmus et al., 2024).

In this work, we formalize two distinct roles that features can play: **input features**, which capture patterns within the model's input, and **output features**, whose main role is to directly influence the tokens the model generates. To find them, we propose **input scores** and **output scores**. First, we obtain a representative set of tokens for each feature by applying the logit lens to SAE weights; this projects the weights directly into the vocabulary space (nostalgebraist, 2020; Bloom and Lin, 2024). We define input features as having high input scores—i.e., high overlap between their top-activating tokens and top logit lens tokens. We de-

[*]Work partially done during an internship at Amazon.
[1]Our code is available at https://github.com/technion-cs-nlp/saes-are-good-for-steering.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 10253–10271
November 4-9, 2025

Layer: 8 Feature 9508

Top-5 Logit Lens Tokens: [PRIMARY, _PRIMARY, _Primary, _primary, Primary]

Input Score: 0.822

Output Score: $5*10^{-6}$

Steering Factor	Generated Text	Effect
Low	*"She saw a woman in the distance, a woman who was dressed in black clothes."*	No Steering Effect
Medium	*"She saw a teacher who was responsible for teaching her son."*	No Steering Effect
High	*"She saw a school school school school school school education school school school school school primary school school school school primary school"*	Generated as if the previous word was "primary"

(a) Steering with an input feature.

Layer: 22 Feature 8827

Top-5 Logit Lens Tokens: [_hair, _Hair, _hairs, hair, Hair]

Input Score: 0

Output Score: **0.808**

Steering Factor	Generated Text	Effect
Low	*"She saw a woman in the distance, a woman who was dressed in black clothes."*	No Steering Effect
Optimal	***"She saw a girl in the crowd, her hair a mess, but her hair was hers"***	**Coherent steered text** ✓
High	*"She saw a hair hair hair hair hair hair hair hair hair hair hair hair hair hair hair hair hair hair hair hair"*	Repeated generation of the logit lens tokens

(b) Steering with an output feature.

Figure 2: **Examples of steering with input and output features. (a)** An input feature, which activates strongly on tokens like "_primary" (leading to a high input score of 0.82), fails to steer generation meaningfully; with a high steering factor, the model degenerates into repeating the token "school", as if continuing from the word "primary". **(b)** An output feature, with an output score of 0.81, yields meaningful, coherent generations when steered at an optimal steering factor.

fine output features as having high output scores—i.e., intervening on the feature increases the probability of its top logit lens tokens in the final output distribution. Notably, the input score can be computed in parallel for all features over a general dataset; the output score requires only one forward pass and no concept-specific data. We quantitatively show that these roles rarely co-occur and tend to emerge at different layers in the model. Specifically, features in earlier layers primarily act as detectors of input patterns, while features in later layers are more likely to drive the model's outputs, consistent with prior analyses of LLM neuron functionality (Lad et al., 2024; Marks et al., 2025).

By calculating the input and output scores of features extracted from Gemma-2 (2B and 9B) (Team et al., 2024), we show that features with high output scores are more effective for coherent and high-quality steering. This yields a practical feature selection method, illustrated in Figure 1: starting with a (typically small) set of candidate features for steering, our scores are computed over this set to select a more effective subset. Figure 2 demonstrates the difference when steering with input features vs. output features: steering with a feature that has a high input score but low output score fails to meaningfully influence the generation. In contrast, steering using a feature with high output score and low input score yields better steering, as well as more fluent and semantically coherent completions.

We demonstrate the effectiveness of these insights on the recent AxBench (Wu et al., 2025), a benchmark for evaluating steering methods. While AxBench found SAEs to be poor for steering, our feature selection results in a 2–3x improvement, causing SAE steering (an unsupervised method) to score significantly closer to supervised methods like LoRA (Xu et al., 2024).

In summary, our contributions are threefold:

- We propose a taxonomy of features according to whether they are more sensitive to analyzing the input or affecting the output, and propose ways of categorizing features into these different roles.
- We propose a practical method for finding features effective for steering.
- Using our results, we engage with current debates on the utility of SAEs for steering, and characterize why these approaches did not find strong results.

2 Preliminaries

2.1 Sparse Autoencoders

Sparse Autoencoders (SAEs) were recently proposed as a method to address the problem of **polysemanticity**, where individual neurons entangle multiple unrelated concepts, and where a single concept may be distributed across many neurons (Bricken et al., 2023). Given a hidden representation $x \in \mathbb{R}^n$, an SAE consists of an encoder and a

decoder, defined as:

$$a(x) := \sigma(W_{\text{enc}}x + b_{\text{enc}}), \quad (1)$$
$$\hat{x}(a) := W_{\text{dec}}a + b_{\text{dec}}. \quad (2)$$

where $W_{\text{enc}}, W_{\text{dec}}, b_{\text{enc}}, d_{\text{dec}}$ are trainable parameters of the SAE.

The encoder maps the latent x into a higher-dimensional sparse vector $a(x)$, or a for short, which we refer to as the activations. The decoder reconstructs x from a as a sparse linear combination of the learned features, given by the columns of W_{dec}. Sparsity and non-negativity of the activations are enforced through the non-linearity σ, often JumpReLU (Rajamanoharan et al., 2024), and regularization.

2.2 The Logit Lens

The logit lens is a widely used interpretability tool for analyzing the hidden representations of language models (nostalgebraist, 2020). Given a hidden state $x \in \mathbb{R}^n$ at any layer of the model, the logit lens passes x through the final layer norm, LN, then projects x onto the vocabulary space by applying the unembedding matrix $W_{\text{unembed}} \in \mathbb{R}^{n \times |\mathcal{V}|}$, where $\mathcal{V}$ is the model's vocabulary. This produces a vector of predicted logits:

$$\ell(x) := W^{\top}_{\text{unembed}}(\text{LN}(x)). \quad (3)$$

The resulting logits $\ell(x)$ can be interpreted as the model's token predictions. We denote the top-k predicted tokens as $\ell(x)_k$.

Recent work has demonstrated that the logit lens can be applied not only to hidden representations, but also model weights (Dar et al., 2023), gradients (Katz et al., 2024), and even in multi-modal settings (Toker et al., 2024).

Bloom and Lin (2024) suggested applying the logit lens to SAE feature weights as a way to interpret their roles. To interpret an SAE feature f_i, corresponding to W^i_{dec}, the i-th column in the decoder matrix, we compute:

$$\ell(f_i) = W^{\top}_{\text{unembed}}\left(\text{LN}\left(W^i_{\text{dec}}\right)\right) \quad (4)$$

Following this body of work that views the logit lens as informative explanations to models' computations and weights, we view $\ell(f_i)_k$ as a faithful explanation of the feature's role. We use $k = 20$, and denote this list as ℓ for brevity.

2.3 Steering LMs

We define **steering** as influencing the output of an LM towards a desired concept. Successful steering should maintain the quality and coherency of the generated text. In other words, we seek a *minimal* change to a model's computation that adds or subtracts a concept's influence. Steering can be done by various methods, recently using SAEs (Durmus et al., 2024; Wu et al., 2025).

Formally, given a model M and a prompt x, we obtain a steered text $\tilde{y}$ by applying an intervention $\Phi(\cdot)$ on some intermediate representation h:

$$\tilde{y} = M_{h \leftarrow \Phi(h)}(x) \quad (5)$$

Similarly to Templeton et al. (2024), in order to steer an LM towards a concept c encoded in an SAE feature f_i at layer l, we define Φ as follows: first, we pass a prompt prefix p through the model. At layer l, we pass the latent representation x^l through the SAE encoder to obtain the activations vector, a. We record the max-activating feature, denoted a_{max}. Then, we obtain a new activation vector using steering factor s:

$$\tilde{a} = \begin{cases} a_j & j \neq i \\ a_j + s \cdot a_{\text{max}} & j = i \end{cases} \quad (6)$$

We pass the steered activation vector through the SAE decoder, to obtain $\Phi(x^l) = W_{\text{dec}}\tilde{a} + b_{\text{dec}}$. , and continue as usual with the rest of the forward pass.

Similarly to Aleksandar Makelov (2024), to evaluate steering success we measure the generation success w.r.t. ℓ_k by calculating the number of appearances of any token in ℓ_k in the generated text. Given a set of sentences S:

$$\text{Gen Success@k}(S) = \frac{\sum_{s \in S} |\{t \in s \mid t \in \ell_k\}|}{|S|} \quad (7)$$

Additionally, we use perplexity measured using Gemma-2-9B to quantify the generation coherence of the entire generated text.

3 Feature Roles Across Layers

In this section, we explore how features specialize across different layers by examining their relationship to the model's input and output tokens. We first define input and output scores that measure the relationship of a feature with the model's inputs or outputs. Then, we describe experiments that report these scores across model layers.

3.1 Input Features

An input feature is a feature whose behavior is closely tied to the tokens that activate it. Intuitively, if a feature consistently activates on a particular set of tokens, and its logit lens representation reflects the same tokens, then it is likely capturing information directly from the input.

Input Score. Given a large corpus, for each feature, let S denote a set of sentences where the feature activated strongly on some tokens in each sentence. For each sentence, we find the maximally activated token. Let T denote the set of top activated tokens across all sentences in S. The input score is the fraction of the top activated tokens that are found in ℓ, the top tokens when projecting the feature with the logit lens:

$$S_{in} = \frac{|\{t \in T \mid t \in \ell\}|}{|T|} \tag{8}$$

In practice, we use the pre-computed activations from Neuronpedia (Lin, 2023) to obtain the sentences S. We verify that S has at least 20 sentences and take a maximum of 100 sentences per feature.

3.2 Output Features

Output features were first mentioned by Paulo et al. (2024) as features whose effect on the model's output can be easily explained in natural language. Natural language explanations predispose us to errors in both precision *and* recall (Huang et al., 2023); therefore, given a target concept, we quantify steering quality as consistency between the set of logit lens tokens and the set of tokens that the steering operation promotes.

Output Score. To measure the effect of a feature on the model's output distribution, we perform an intervention during a forward pass and evaluate the change in the rank and probability assigned to tokens in ℓ. We first use a neutral prompt ($x =$ "In my experience,") to obtain the model's prior distribution over token ranks and probabilities. Then, we intervene on the feature's activation value using a large steering factor (we use 10), as in Equation (6).

We record the ranks of the tokens in ℓ and their probabilities; we denote the token with the highest rank as ℓ^*, its rank as $r(\ell^*)$, and its probability as $p(\ell^*)$. The output score is then the difference in rank-weighted probabilities between the original and counterfactual output distributions:

$$P(\mathcal{M}) = (1 - \frac{r(\ell^*, \mathcal{M})}{|V|}) p(\ell^*, \mathcal{M}) \tag{9}$$

$$S_{out} = P(\mathcal{M}_{h \leftarrow \Phi(h)}) - P(\mathcal{M}) \tag{10}$$

where V is the model's vocabulary. If x is a neutral prompt, then $S_{out} \propto P(\mathcal{M}_{h \leftarrow \Phi(h)})$, so we can compute S_{out} quickly using a single forward pass by only computing the rank-weighted probability after the intervention. This score is robust to the specific choice of neutral prompt; see details in Appendix D.

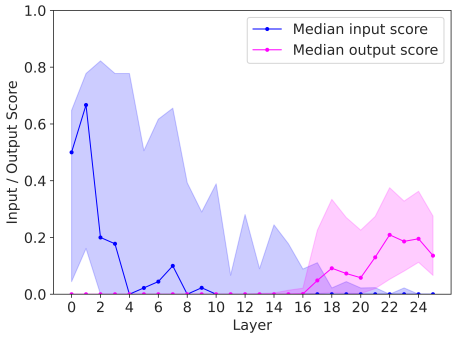
3.3 Experimental Setup

We focus our analysis on Gemma-2 (2B and 9B) using the Gemma-Scope 16K SAEs (Team et al., 2024; Lieberum et al., 2024), Llama-3.1 8B with Llama-Scope SAEs (Grattafiori et al., 2024; He et al., 2024), and Pythia-70m (Biderman et al., 2023; Huben et al., 2024). Our analysis spans 100 features randomly sampled from each layer. For Pythia-70m we limited our sampling to features with at least 10 recorded activations, since many features do not have any recorded activations on Neuronpedia.

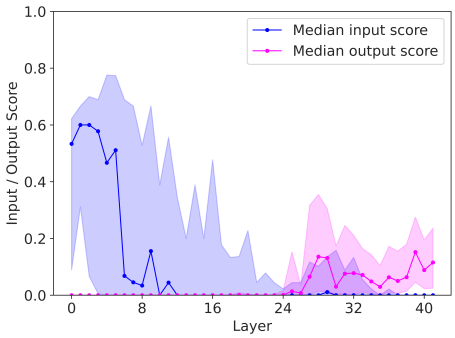
3.4 Results

Figure 3 shows the distribution of input and output scores across layers for Gemma-2-2B and Gemma-2-9B. In early layers (0–50% of model depth), features tend to have high input scores and near-zero output scores, suggesting they are predominantly input-aligned. Later layers (66–100% of model depth) show an opposite trend: input scores drop to near-zero, while output scores increase significantly. These later-layer features no longer reflect the tokens they are activated on, but instead align with the tokens they promote in the model's output, indicating a shift toward output-aligned behavior. Interestingly, middle layers exhibit low scores for both metrics, suggesting that these features may play intermediate roles that are neither purely input-aligned nor strongly output-promoting.

For Llama-3.1 we do not observe any trend in early layers of the model (Appendix B), likely due to limitations of applying the logit lens to early layer representations (nostalgebraist, 2020). At around 50% of the model's depth we begin to observe non-zero values for both scores, with low values of input scores and increasingly growing values of output scores as layers progress, as in the Gemma-2 results.



(a) Gemma-2-2B.

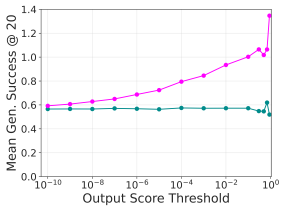


(b) Gemma-2-9B.

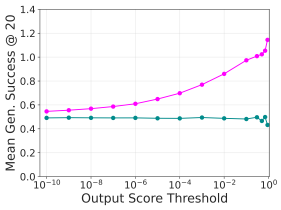
Figure 3: **Input and output scores across layers in Gemma-2-2B and Gemma-2-9B.** The solid lines represent the median input score (blue) and output score (magenta), while the shaded regions denote the interquartile range (25th to 75th percentile), capturing the variability across features within each layer. Early layers are characterized by features with high input scores, while high output scores emerge in later layers.

For Pythia, we observe a slightly different trend (Appendix B). While output score gradually increases around 50% of the model's depth as expected, the input score is mainly zero for most of the tested features. This may be due to the fact that this model is significantly smaller compared to the other models we examined (70 million parameters compared to 2–9 billion), which may lead it to parse and encode information differently within its latent space.

Interestingly, and unlike Llama-3.1, early layers in both Gemma models and Pythia seem to be interpretable with the logit lens. We find that many features promote a coherent and human-understandable set of tokens, as reflected by high input scores as early as layers 0 and 1. (See Appendix C for examples.)



(a) Gemma-2-2B.



(b) Gemma-2-9B.

Figure 4: Magenta indicates the mean generation success@20 when filtering out features with output scores below different thresholds. Green indicates the mean generation success@20 after filtering randomly sampled sets of features of the same size. Filtering results in significant increase in generation success.

4 Identifying Features for Steering

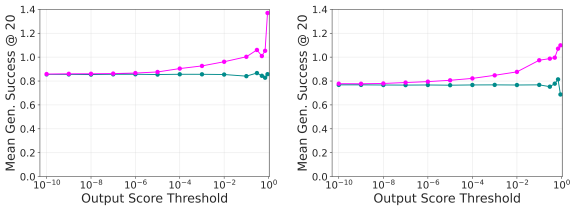
The output score measures the alignment between the effects of SAE features on the model's output distribution and the expected set of tokens—in our case, their top logit lens tokens. In this section we hypothesize that features with high output scores are more effective for steering. To test this, we evaluate generation success when filtering out features with low output scores at different thresholds.

4.1 Experimental Setup

We use 50 prompt prefixes and generate up to 20 tokens, obtaining 50 generated texts for each feature (more details in Appendix E). For each feature we calculate the mean generation success across the generated texts, and filter out steering factors leading to generation success greater than 3. Intuitively, the generation success measures the rate in which the model generates concept-related tokens. Based on early experiments, we find that 3 is a good upper value that balances steering and coherence. We choose the optimal steering factor as the one that maximizes $\frac{\text{Gen Success@20}}{\text{Perplexity}}$, where we normalize both metrics to a 0–1 range by dividing with the maximum value across all data samples.

4.2 Qualitative Results

Figure 2 demonstrates steering with two features from Gemma-2-2B: one having a high input score and low output score (an input feature), and the other, an output feature, having a high output score and low input score. Steering using the input feature fails to meaningfully influence the generation. When using a high steering factor, this results in repeatedly generating tokens related to the feature's activation, as if continuing from the word "primary". In contrast, steering using a feature with



(a) Gemma-2-2B. (b) Gemma-2-9B.

Figure 5: Even in later layers of the model (16–25 for Gemma-2-2B and 24–41 for Gemma-2-9B), filtering features with low output scores increases mean generation success. Magenta: filtering by output scores. Green: filtering random sets of features of the same size.

high output score and low input score using an optimal steering factor yields fluent and semantically coherent completions. Additional examples are shown in Appendix E.

4.3 Quantitative Results

Figure 4 shows the mean generation success @ 20 of steered generations when filtering out features with output scores below varying thresholds (magenta) on Gemma-2-2B and Gemma-2-9B. As the threshold increases, performance improves steadily, indicating that features with higher output scores consistently lead to more successful steering. We observe an increase in the mean generation success score from around $0.5 - 0.6$ for both models without any filtering, to 1.1–1.4 using a threshold of 0.9. A threshold of 0.01 is sufficient for filtering out about 60% of the features, increasing the mean generation success by around 0.4 points. We compare this against a random baseline (green): filtering randomly sampled subsets of features of the same size does not lead to any significant improvements (results are average of 10 random samples per subset size). Llama-3.1 and Pythia show similar trends; see Appendix B.

The results in Section 3 suggest that features with high output scores occur predominantly in later layers. Figure 5 shows the generation success when filtering based on the output score, evaluated only on features from later layers of the model: 16–25 for Gemma-2-2B and 24–41 for Gemma-2-9B. Taking only features from later layers, Gemma-2-2B and Gemma-2-9B achieve generation success scores of about 0.8. By considering only top-scoring features, the mean generation success increases to around 1.1–1.4 for both models. These results show that even within these later layers, the output score is a useful tool for filtering out features that lead to poor steering results.

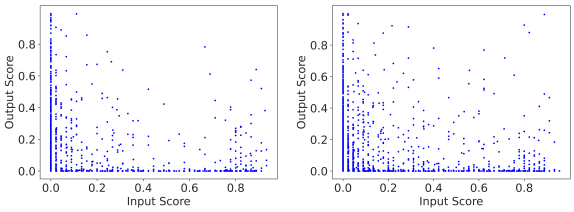
		Gemma-2-9B-it	
		L20	L31
Our results	$\text{SAE}_{(S_{out} \geq 0.1)}$	0.546	<u>0.470</u>
	$\text{SAE}_{(S_{out} \geq 0.01)}$	0.338	0.454
	$\text{SAE}_{(S_{out} \geq 0.001)}$	0.373	0.415
	$\text{SAE}_{(S_{out} \geq 0.0001)}$	0.325	0.401
	$\text{SAE}_{\text{(No Filter)}}$	0.293	0.387
AxBench reported results (Wu et al. 2025)	Prompt	**1.075**	**1.072**
	LoReFT	0.777	0.764
	LoRA	0.602	0.580
	ReFT-r1	<u>0.630</u>	0.401
	DiffMean	0.322	0.158
	SAE	0.191	0.140
	SAE-A	0.186	0.143

Table 1: **Results on the Concept500 dataset from AxBench on instruction-tuned Gemma-2-9B.** (Top) Results when steering with SAEs after filtering out features with S_{out} lower than different thresholds. (Bottom) Results reported by Wu et al. (2025). **Bold** indicates the best score, <u>underline</u> indicates the best score among representation-based methods. Grey indicates non-representation-based methods. After filtering based on output scores, SAEs achieve the best score among representation-based methods at L31, and reach 90.7% of the best method's performance at L20.

4.4 Evaluation on AxBench

AxBench was recently proposed as a dataset to evaluate steering methods (Wu et al., 2025). They compare steering with SAE features to other methods (including supervised methods), and find SAE features relatively ineffective. However, we believe this is partially due to a non-principled selection of SAE features; we propose to remedy this using the output score.

We evaluate our findings on instruction-tuned Gemma-2-9B using the Concept500 dataset of Wu et al. (2025), which includes 1000 ⟨concept, SAE feature⟩ pairs from layers 20 and 31 of the model. As in AxBench, we randomly sample 10 instructions for each concept-feature pair from instruction datasets aligned with the concept's genre. Five are used to select the optimal steering factor (as detailed in Section 4.1), and the remaining five are used exclusively for evaluation. We evaluate the steered texts using the metrics defined by Wu et al.: (1) the concept score measures if the concept was incorporated in the generated text; (2) the fluency score measures the coherency of the text; and (3) the instruct score measures the alignment of the generated text with the given instructions. For each feature, we compute the harmonic mean



(a) Gemma-2-2B. (b) Gemma-2-9B.

Figure 6: **Relationship between input and output scores for features in Gemma-2-2B and Gemma-2-9B.** Most features lie near the axes, indicating that most features are either input *or* output features—though a few are both.

of the three metrics. See Appendix F for additional details on the steering setup, metrics, and baseline methods.

We report the mean score on layers 20 and 31 of instruction-tuned Gemma-2-9B (Table 1). We find a nearly threefold improvement in SAE steering scores relative to Wu et al.. Note that our replication of their experimental setting (without filtering) yields higher scores for SAEs; this can be a result of different sampled instructions per feature,[2] or possibly due to evaluation instability introduced by the use of an external LLM. With output score filtering, SAE features top steering performance among the representation-based methods at L31; for L20, they get 90.7% of the performance of the best method. This is in contrast with the results of Wu et al., where SAEs significantly underperforms ReFT-r1, a weakly supervised method they propose as a competitive alternative to prompting. These results demonstrates that with effective feature selection SAEs are comparable with existing methods, including supervised or weakly-supervised methods which require concept-specific datasets.

5 The Relationship Between Input and Output Scores

We next examine how input and output scores interact. Figure 3 suggests that high input and output scores are rarely observed in the same layers, but do they sometimes co-occur in the same features?

Figure 6 visualizes the relationship between the two scores: Indeed, most samples cluster near the axes, exhibiting either a high output score with a near-zero input score, or vice-versa. However, there do exist hybrid features: features with both high input and high output scores.

Figure 7 illustrates generation results when steering with hybrid features from Gemma-2-2B. Generated tokens that also appear in the feature's top-20 logit lens tokens are highlighted in magenta. These are often not the top-ranked tokens under the logit lens, but they tend to rank moderately high and collocate with the top tokens (highlighted in blue). For instance, for feature 6820 in layer 18, "contact" is the top logit lens token, but the output text repeatedly includes "lenses", a token that appears further down the list but that is semantically and syntactically related.

We verify this intuition by quantifying collocation patterns between generated tokens and top logit lens tokens using their pointwise mutual information (PMI)[3]. A PMI of zero indicates that two tokens co-occur no more frequently than chance, and negative and positive scores indicate lower- or higher-than-chance co-occurrence, respectively.

Features with high output score ($S_{out} \geq 0.1$) and low input score ($S_{in} < 0.1$) have a negative PMI value on average (Figure 8). This can be attributed to the high generation success of these features, i.e., that the top logit lens token is equal to the generated token. In most cases, two instances of the same token are not likely to consecutively co-occur, thus obtaining a low PMI score.

Importantly, features that have a high input score ($S_{in} \geq 0.1$) tend to generate tokens with significantly higher PMI relative to their top logit lens tokens, regardless of their output score values. This suggests that hybrid features may be less favorable for steering, despite their high output score.

6 Related Work

6.1 Stages of Processing in LMs

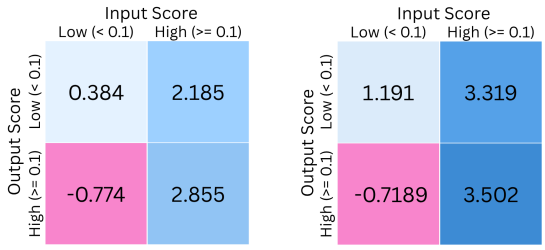
The different stages of processing within NLP models have long been studied (Belinkov et al., 2017; Zhang and Bowman, 2018; Liu et al., 2019; Brunner et al., 2020). In transformer-based LMs, a large body of work shows that different properties emerge in different layers. Early layers focus on syntactic tasks such as POS, while semantic information appears in later layers (Tenney et al., 2019; Elazar et al., 2021; Geva et al., 2021). More recent work has demonstrated that intermediate layers are

[2] Wu et al. (2025) sample 10 instructions per feature from pre-existing datasets, but do not release these instructions. We sample 10 instructions from the same datasets, which may be different compared to the sample of Wu et al..

[3] We use the pre-computed PMI over the webtext corpus in NLTK (Bird and Loper, 2004), and only include token pairs that have this pre-computed score (100–200 pairs per model).

Layer	Feature	Input Score	Output Score	Top-20 Logit Lens Tokens	Generated Text
3	6824	0.911	0.126	[have, _has, have, _Have, _had, _HAVE, Have, _Has, has, _been, Has, HAVE, _HAS, _telah, _έχουν, έχει, _hebben, _hav, _hanno, _heeft]	I believe the answer been found so far.
13	9105	0.778	0.387	[entered, _enter, Entries, _enters, _entering, _Entries, _Entry, _Entering, Enter, _Enter, _ENTER, _entry, entered, Entry, Entering, _ENTRY, _Entered, _entries, _into, enter]	She saw a into a room, the room into the hall, the hall into the back of the house
17	15607	0.867	0.354	[LONG, _Long, getLong, _long, Long, long, LONG, _Longo, _longs, _Lasting, _longa, _lasting, _ago, _longer, _durée, _longue, _term, _panjang, _Longer, _length]	It is observed that the term "term-term memory-term" has been used in the article
18	6820	0.867	0.215	[contact, _Contact, CONTACT, _CONTACT, contact, _Contacts, _contacts, Contact, Contacts, contacts, _kontak, _Kontak, _pinulongan, Kontakt, _kontakt, CONTACTS, _contacto, _lenses, _Contacto, _Kontakt]	The news mentioned that the lenses of the lenses of the lenses with the lenses of the lenses ...
20	448	0.867	0.299	[_missions, _Missions, Missions, _mission, _Mission, _MISSION, _statement, Mission, _Impossible, mission, _Statement, _Viejo, _misi, _impossible, _cumplido, _missão, missions, _imposible, _mision, statement]	It is observed the statements, in statement 0 statement 1, is false

Figure 7: Generation results when steering with features that have both high input and high output scores in Gemma-2-2B. The top logit lens tokens (left; top-1 in blue) do not appear directly in the generated text (right). The steered tokens (magenta), appearing lower in the logit lens ranking, often have strong collocational associations with the top logit lens tokens.



(a) Gemma-2-2B. (b) Gemma-2-9B.

Figure 8: Pointwise Mutual Information (PMI) between generated tokens and top-1 logit lens tokens, grouped by input/output score thresholds. Features with high output scores and low input scores (magenta) tend to have negative PMI values, likely due to exact token repetition during generation, which indicates successful steering. In contrast, features with high input scores (blue) consistently yield higher PMI values, indicating stronger collocational relationships with their top tokens.

responsible for retrieving factual knowledge and enriching latent representations (Meng et al., 2022; Geva et al., 2023; Hernandez et al., 2024; Arad et al., 2024).

Another line of work has focused on so-called prediction neurons, which increase the probability of coherent sets of tokens. This work characterizes prediction neurons by properties of their logit lens distribution (Gurnee et al., 2024; Lad et al., 2024; Bloom and Lin, 2024). In contrast, our output score directly measures the *causal* effect of a feature on predicting a pre-defined set of tokens via counterfactual interventions. Relatedly, Lad et al. (2024) have shown that neurons in early layers pay more attention to input tokens in their proximity compared to later layers, while prediction neurons emerge later, after about 50% of the model depth, in line with our findings on SAEs.

A closely related line of work aims to explain SAE features in natural language—for instance, by feeding inputs and activations into an external LLM (Bills et al., 2023; Huben et al., 2024). However, this method results in errors in both precision and recall (Huang et al., 2023), and *negatively* (albeit weakly) correlates with their causal role on average (Paulo et al., 2024). Gur-Arieh et al. (2025) suggest that SAE features are better explained in terms of their activations *and* their effect on the output(as quantified by projecting features into vocabulary space). In this work, our aim is not to *explain* features, but rather *categorize* them with respect to their usefulness for steering. Additionally, we differentiate between two key feature roles (often mutually exclusive); this helps explain these prior findings and failure cases.

6.2 Steering LMs

Many approaches exist for precisely influencing the outputs generated by LMs (Zou et al., 2023). These include prompt engineering (Wu et al., 2025; Taveekitworachai et al., 2024), steering vectors (Subramani et al., 2022; Teehan et al., 2022; Liu et al., 2023) or inference-time interventions on activations (Turner et al., 2023; van der Weij et al., 2024; Rimsky et al., 2024). Steering was shown to be useful not only for directing generated content to a specific topic or concept, but also for style transfer (Lai et al., 2024), mitigating hallucinations (Li et al., 2023a; Simhi et al., 2024), and debiasing (Li et al., 2025).

While most work in this area steers via interven-

tions on full hidden states, earlier work attempted to influence model behavior by intervening on small sets of neurons (Bau et al., 2019). However, the polysemanticity of neurons makes them poor candidates for effective steering (Bricken et al., 2023). In contrast, SAEs were shown to result in meaningful steering towards human-understandable concepts; a famous example involved steering toward responses related to the Golden Gate Bridge (Templeton et al., 2024), and another involved amplifying or mitigating social and political biases (Durmus et al., 2024; Marks et al., 2025). Recently, Wu et al. (2025) evaluated SAE steering against many methods, including supervised methods such as full fine-tuning, prompting, and difference-in-means (Larsen et al., 2016). They found that even these simple baselines outperform SAEs. However, our work shows that most of the gap can be closed via more careful choice of SAE features. Typical work chooses features for steering based on natural language explanations generated based on each feature's activation patterns (Huben et al., 2024; Durmus et al., 2024); our findings instead suggest that *influence on output* is a better proxy for steering efficacy, and that input activations have little predictive power for finding good steering features.

7 Conclusions

We have formalized and analyzed two roles demonstrated by sparse autoencoder (SAE) features. We have defined the notion of an input score, which captures the alignment of a feature's activations with its top logit lens tokens, and an output score, which quantifies the alignment of the top logit lens tokens with the feature's effect on the model's generations. We demonstrate that features with high output scores are significantly more effective for steering, whereas features with high input scores are relatively ineffective, even when they appear relevant to the steering concept.

Limitations

While our work provides an efficient framework for identifying and leveraging output features for generation steering, several limitations remain. First, our analysis is restricted to features extracted from the residual stream, and does not account for features derived from other components such as attention or MLP layers. As a result, our taxonomy may not capture the full range of functional roles present across the model.

Additionally, our method focuses on steering using a single SAE feature. In practice, interactions between features may lead to better and more complex effects on generation (Wattenberg and Viégas, 2024; Singhvi et al., 2025). Understanding how multiple features combine or interfere remains an open challenge.

Ethical Considerations

Our work suggests a framework that improves one's ability to choose meaningful SAE features for steering LMs. While steering can support positive use cases such as controllable text generation, personalization, and bias mitigation, it can also introduce risks that must be considered. In particular, steering methods can be used to manipulate model outputs in ways that circumvent safety mechanisms or amplify harmful content. Additionally, our methods rely on pre-trained models that may contain biases or harmful associations. Although our framework can help isolate and suppress such patterns, it can also be misused to reinforce them.

Acknowledgments

This research was supported by an Azrieli Foundation Early Career Faculty Fellowship and Open Philanthropy. Dana Arad is supported by the Ariane de Rothschild Women Doctoral Program. Aaron Mueller was supported by a postdoctoral fellowship under the Zuckerman STEM Leadership Program. This research was funded by the European Union (ERC, Control-LM, 101165402). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

We thank Shoval Lagziel and Yonatan Aflalo for early feedback on this work. We thank Adi Simhi for her support and feedback.

References

Nathaniel Monson Aleksandar Makelov. 2024. Evaluating sparse autoencoders for controlling open-ended text generation. In *Second NeurIPS Workshop on Attributing Model Behavior at Scale*.

Anthropic. 2024. The claude 3 model family: Opus, sonnet, haiku. https://assets.anthropic.com/m/61e7d27f8c8f5919/

original/Claude-3-Model-Card.pdf. Accessed: 2025-04.

Dana Arad, Hadas Orgad, and Yonatan Belinkov. 2024. Refact: Updating text-to-image models by editing the text encoder. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2537–2558.

Anthony Bau, Yonatan Belinkov, Hassan Sajjad, Nadir Durrani, Fahim Dalvi, and James Glass. 2019. Identifying and controlling important neurons in neural machine translation. In *International Conference on Learning Representations*.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James R. Glass. 2017. What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 861–872. Association for Computational Linguistics.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR.

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

Steven Bird and Edward Loper. 2004. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 214–217, Barcelona, Spain. Association for Computational Linguistics.

Joseph Bloom and Johnny Lin. 2024. Understanding sae features with the logit lens.

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*. Https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Gino Brunner, Yang Liu, Damian Pascual, Oliver Richter, Massimiliano Ciaramita, and Roger Wattenhofer. 2020. On identifiability in transformers. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. 2023. Free dolly: Introducing the world's first truly open instruction-tuned llm.

Guy Dar, Mor Geva, Ankit Gupta, and Jonathan Berant. 2023. Analyzing transformers in embedding space. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 16124–16170. Association for Computational Linguistics.

Esin Durmus, Alex Tamkin, Jack Clark, Jerry Wei, Jonathan Marcus, Joshua Batson, Kunal Handa, Liane Lovitt, Meg Tong, Miles McCain, Oliver Rausch, Saffron Huang, Sam Bowman, Stuart Ritchie, Tom Henighan, and Deep Ganguli. 2024. Evaluating feature steering: A case study in mitigating social biases.

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

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. 2023. Dissecting recall of factual associations in auto-regressive language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 12216–12235. Association for Computational Linguistics.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 5484–5495. Association for Computational Linguistics.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Yoav Gur-Arieh, Roy Mayan, Chen Agassy, Atticus Geiger, and Mor Geva. 2025. Enhancing automated interpretability with output-centric feature descriptions. *CoRR*, abs/2501.08319.

Wes Gurnee, Theo Horsley, Zifan Carl Guo, Tara Rezaei Kheirkhah, Qinyi Sun, Will Hathaway, Neel Nanda, and Dimitris Bertsimas. 2024. Universal neurons in GPT2 language models. *Trans. Mach. Learn. Res.*, 2024.

Zhengfu He, Wentao Shu, Xuyang Ge, Lingjie Chen, Junxuan Wang, Yunhua Zhou, Frances Liu, Qipeng Guo, Xuanjing Huang, Zuxuan Wu, et al. 2024. Llama scope: Extracting millions of features from llama-3.1-8b with sparse autoencoders. *arXiv preprint arXiv:2410.20526*.

Evan Hernandez, Arnab Sen Sharma, Tal Haklay, Kevin Meng, Martin Wattenberg, Jacob Andreas, Yonatan Belinkov, and David Bau. 2024. Linearity of relation decoding in transformer language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Jing Huang, Atticus Geiger, Karel D'Oosterlinck, Zhengxuan Wu, and Christopher Potts. 2023. Rigorously assessing natural language explanations of neurons. In *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 317–331, Singapore. Association for Computational Linguistics.

Robert Huben, Hoagy Cunningham, Logan Riggs, Aidan Ewart, and Lee Sharkey. 2024. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Shahar Katz, Yonatan Belinkov, Mor Geva, and Lior Wolf. 2024. Backward lens: Projecting language model gradients into the vocabulary space. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pages 2390–2422. Association for Computational Linguistics.

Vedang Lad, Wes Gurnee, and Max Tegmark. 2024. The remarkable robustness of llms: Stages of inference? *CoRR*, abs/2406.19384.

Wen Lai, Viktor Hangya, and Alexander Fraser. 2024. Style-specific neurons for steering llms in text style transfer. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 13427–13443.

Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. 2016. Autoencoding beyond pixels using a learned similarity metric. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1558–1566, New York, New York, USA. PMLR.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023a. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36:41451–41530.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023b. Alpacaeval: An automatic evaluator of instruction-following models. `https://github.com/tatsu-lab/alpaca_eval`.

Yichen Li, Zhiting Fan, Ruizhe Chen, Xiaotang Gai, Luqi Gong, Yan Zhang, and Zuozhu Liu. 2025. Fairsteer: Inference time debiasing for llms with dynamic activation steering. *arXiv preprint arXiv:2504.14492*.

Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, János Kramár, Anca Dragan, Rohin Shah, and Neel Nanda. 2024. Gemma scope: Open sparse autoencoders everywhere all at once on gemma 2. *arXiv preprint arXiv:2408.05147*.

Johnny Lin. 2023. Neuronpedia: Interactive reference and tooling for analyzing neural networks. Software available from neuronpedia.org.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 1073–1094. Association for Computational Linguistics.

Sheng Liu, Haotian Ye, Lei Xing, and James Zou. 2023. In-context vectors: Making in context learning more effective and controllable through latent space steering. *arXiv preprint arXiv:2311.06668*.

Samuel Marks, Can Rager, Eric J Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. 2025. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. In *The Thirteenth International Conference on Learning Representations*.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

nostalgebraist. 2020. Interpreting GPT: The logit lens. lesswrong, 2020.

Kyle O'Brien, David Majercak, Xavier Fernandes, Richard Edgar, Jingya Chen, Harsha Nori, Dean

Carignan, Eric Horvitz, and Forough Poursabzi-Sangde. 2024. Steering language model refusal with sparse autoencoders. *arXiv preprint arXiv:2411.11296*.

Gonçalo Paulo, Alex Mallen, Caden Juang, and Nora Belrose. 2024. Automatically interpreting millions of features in large language models. *arXiv preprint arXiv:2410.13928*.

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

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. 2024. Steering llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15504–15522.

Adi Simhi, Jonathan Herzig, Idan Szpektor, and Yonatan Belinkov. 2024. Constructing benchmarks and interventions for combating hallucinations in llms. *arXiv preprint arXiv:2404.09971*.

Divyansh Singhvi, Diganta Misra, Andrej Erkelens, Raghav Jain, Isabel Papadimitriou, and Naomi Saphra. 2025. Using shapley interactions to understand how models use structure. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 20727–20737, Vienna, Austria. Association for Computational Linguistics.

Nishant Subramani, Nivedita Suresh, and Matthew E Peters. 2022. Extracting latent steering vectors from pretrained language models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 566–581.

Pittawat Taveekitworachai, Febri Abdullah, and Ruck Thawonmas. 2024. Null-shot prompting: rethinking prompting large language models with hallucination. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 13321–13361.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Ryan Teehan, Miruna Clinciu, Oleg Serikov, Eliza Szczechla, Natasha Seelam, Shachar Mirkin, and Aaron Gokaslan. 2022. Emergent structures and training dynamics in large language models. In *Proceedings of BigScience Episode# 5–Workshop on Challenges & Perspectives in Creating Large Language Models*, pages 146–159.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, Alex Tamkin, Esin Durmus, Tristan Hume, Francesco Mosconi, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter, Chris Olah, and Tom Henighan. 2024. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4593–4601. Association for Computational Linguistics.

Michael Toker, Hadas Orgad, Mor Ventura, Dana Arad, and Yonatan Belinkov. 2024. Diffusion lens: Interpreting text encoders in text-to-image pipelines. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 9713–9728. Association for Computational Linguistics.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J Vazquez, Ulisse Mini, and Monte MacDiarmid. 2023. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*.

Teun van der Weij, Massimo Poesio, and Nandi Schoots. 2024. Extending activation steering to broad skills and multiple behaviours. *arXiv preprint arXiv:2403.05767*.

Martin Wattenberg and Fernanda Viégas. 2024. Relational composition in neural networks: A survey and call to action. In *ICML 2024 Workshop on Mechanistic Interpretability*.

Zhengxuan Wu, Aryaman Arora, Atticus Geiger, Zheng Wang, Jing Huang, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. 2025. Axbench: Steering llms? even simple baselines outperform sparse autoencoders. *CoRR*, abs/2501.17148.

Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D Manning, and Christopher Potts. 2024. Reft: Representation finetuning for language models. *Advances in Neural Information Processing Systems*, 37:63908–63962.

Yuhui Xu, Lingxi Xie, Xiaotao Gu, Xin Chen, Heng Chang, Hengheng Zhang, Zhengsu Chen, Xiaopeng Zhang, and Qi Tian. 2024. Qa-lora: Quantization-aware low-rank adaptation of large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Kelly W Zhang and Samuel R Bowman. 2018. Language modeling teaches you more syntax than translation does: Lessons learned through auxiliary task analysis. *arXiv preprint arXiv:1809.10040*.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, et al. 2023. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*.

A Additional Steering Examples

Tables 2, 3, and 4 show examples of steering Gemma-2-2B, Pythia, and Llama-3.1, respectively. Features with high output scores result in meaningful steering, while features with high input score *and* low output score do not have any visible effect on the generated text.

B Results on Llama-3.1-8B and Pythia-70m

Figure 9 demonstrates the distributions of scores across layers for Llama-3.1 and Pythia.

Figure 10 shows the mean generation success @ 20 of steered generations when filtering out features with output scores below varying thresholds (magenta) on Llama-3.1 and Pythia. Similarly to the Gemma models, as the threshold increases, performance improves steadily, indicating that features with higher output scores consistently lead to more successful steering.

C Identifying Input Features

Appendix C shows examples of features from layers 0 and 1 having high input scores. For each feature, the table includes it's top-5 logit lens tokens as well as examples for input texts that activated this feature. The tokens where the feature activated most strongly are marked using an underline.

D Neutral Prompt Selection

Our output score relies on the use of a single neutral prompt as a prefix to the model's generation. To evaluate the robustness of our score to the specific choice or neutral prompt we randomly selected 10 features from each layer of Gemma-2-2B (240 features overall) and computed their output scores using all 50 of the neutral prompts in Table 6. We find that the correlation between the output score computed with the original prompt and the mean score is 0.9557, which is extremely high and indicates that the exact choice of the prompt has almost no impact on the results.

E Steering Details

For our main experiments, we test steering factor values $s \in \{0.2, 0.4, 0.8, 1.2, 1.6, 2.0, 3.0, 4.0, 6.0, 8.0, 10.0, 20.0\}$. We generate 20 tokens using a temperature of 0.7 after each of the prefixes listed in Table 6.

F AxBench Details

Model Instructions. Each concept in the Concept500 dataset is annotated as either "text", "math", or "code". Following their setup we randomly sample 10 instructions per concept from instruction datasets that match the concept genre: Free Dolly dataset for text instructions (Conover et al., 2023), GSM8K for math (Cobbe et al., 2021), and Alpaca-Eval for code (Li et al., 2023b).

Steering Details. We generate up to 128 tokens per instruction, with a temperature of 0.7, using steering factor values of $\{0.4, 0.8, 1.2, 1.6, 2.0, 3.0, 4.0, 6.0, 8.0, 10.0, 20.0, 40.0, 60.0, 100.0\}$. For each concept, five instructions are used to choose the optimal steering factor (as described in 4.1), and the rest are used for evaluation.

Metrics. We evaluate the steered texts using the metrics defined by Wu et al.: (1) the concept score (cs) measures if the concept was incorporated in the generated text, (2) the fluency score (fs) measures the coherency of the text, and (3) the instruct score (is) measures the alignment of the generated text with the given instructions. For each metric ($m \in \{\text{cs}, \text{fs}, \text{is}\}$) and steered text s, an external rater returns a discrete score of either 0, 1, or 2: $m(s) \in \{0, 1, 2\}$. As the external LLM rater, we use Claude 3.7 sonnet (2025-02-19) (Anthropic, 2024). The prompts for all metrics are given in Tables 7, 8, and9.

For each concept c, we compute each metric over the five test instructions and take the mean: $m(c) = \frac{\sum_{s \in S} m(s)}{|S|}$. The overall score of a concept is the harmonic mean of the three scores: $(\text{cs}(c), \text{fs}(c), \text{is}(c))$.

The cost of obtaining this score for the tested features was 65 USD.

Baseline Methods. Table 1 shows the reported results of steering using various methods, as achieved

Layer	Feature	Input Score	Output Score	Top-5 Logit Lens Tokens	Optimal Steering Factor	Generated Text
11	10662	**0.756**	0	['_engineers', '_engineer', '_engineering', 'Engineers', '_Engineering']	0.2	Funny thing is, I bought my first pair of these shoes (the black leather) over 20 years ago when
13	11961	**0.778**	0	['_machines', 'Machines', 'machines', ' Machines', '_machine']	0.2	A friend of mine once said, "I always wanted to be an architect."
16	731	**0.8**	0	['_exposure', 'exposure', '_Exposure', 'Exposed', '_Exposed']	0.2	Findings show that children in the United States are eating more breakfast foods, but less fruit and vegetables.
18	9085	0.023	**0.142**	[' activism', ' activists', ' activist', ' protest', ' protesting']	6.0	I once heard that the biggest and most powerful movement for human civil activism and peace movement, is an movement for peace movement
19	10015	0.023	**0.734**	['_profile', '_Profile', 'Profile', 'profile', '_PROFILE']	4.0	Findings show that a profile picture in your profile is helpful and makes it more likely that people will add you as a
19	10204	0	**0.451**	['_crime', '_corruption', '_violence', '_fraud', '_crimes']	1.6	Then the man said: "If I commit murder, the crime will be on my conscience."

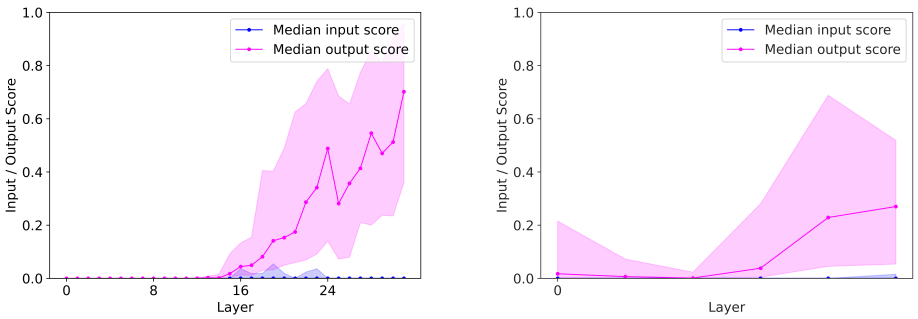
Table 2: Examples of steering with features with different output and input score values in Gemma-2-2B.

Layer	Feature	Input Score	Output Score	Top-5 Logit Lens Tokens	Optimal Steering Factor	Generated Text
4	7772	0	**0.794**	['_Firefox', '_Chrome', '_browser', '_Mozilla', '_browsers']	1.2	Findings show that a site may play a key role in the development of a web browser.
5	21568	0	**0.474**	['_Barack', '_Obama', '_Donald', '_Trump', '_Bush']	0.8	The legend goes that the guy was a great- Barack Obama, he'd probably be the biggest supporter of Barack Obama ever

Table 3: Examples of steering with features with different output and input score values in Pythia-70m.

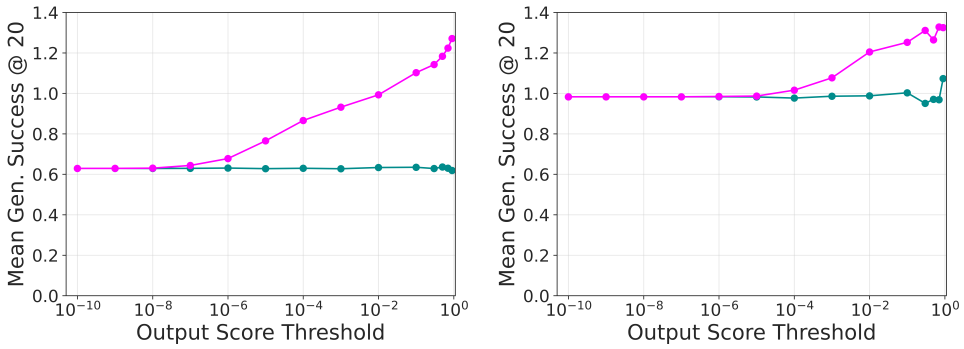
Layer	Feature	Input Score	Output Score	Top-5 Logit Lens Tokens	Optimal Steering Factor	Generated Text
2	25580	**0.436**	0	['_sources', '_SOUR', 'sources', 'iped', '/source']	0.8	That reminds me of the time when I was in a public of knowledge for the answer of information that said the information was said that said
20	17816	0	**0.669**	['_visa', '_immigration', '_Immigration', '_visas', '_Visa']	1.6	I believe that the H-1B visa program is an important tool for employers to access the best talent to fill
26	21627	0	**0.665**	['_disability', '_disabled', '_Disability', '_disable', '_Disabled']	1.2	Findings show that the mainstream media has a strong impact on public opinion on disability.

Table 4: Examples of steering with features with different output and input score values in Llama-3.1-8B.



(a) Llama-3.1-8B. (b) Pythia-70m.

Figure 9: **Input and output scores across layers in Llama-3.1-8B and Pythia-70m.** The solid lines represent the median input score (blue) and output score (magenta), while the shaded regions denote the interquartile range (25th to 75th percentile), capturing the variability across features within each layer. In these models we observe that high output scores emerge in later layers, while input score is mostly zero across all layers.



(a) Llama-3.1-8B. (b) Pythia-70m.

Figure 10: Magenta indicates the mean generation success@20 when filtering out features with output scores below different thresholds. Green indicates the mean generation success@20 after filtering randomly sampled sets of features of the same size. Filtering results in significant increase in generation success.

Layer	Feature	Input Score	Top-5 Logit Lens Tokens	Activated Text
0	725	0.822	['_she', '_It', '_we', 'she', '_OHA']	... euphoria and uncertainty. He asked himself, " ... ... She also recalls the expeditions ... ... with a win. He said he's ...
0	14772	0.889	['inning', 'inb', 'inl', 'IN', 'inare']	... deposition of both elastin and fib rillin ... ... I. Manin, Three-dimensional ... ... ,k\in {1,\dots ...
1	4413	0.844	['cleared', 'cle', ' clearing', ' clearance', ' cleaned']	... set. Once we cleared all the debris, ... clearing room for a future ... ... If this were only cleared away," They said ...
1	3110	0.867	['_deltas', 'Delta', 'DeltaTime', \Delta, 'eltas']	... large potential difference (Δ V = ... atmospheric thickness, Δz_eff, ... finite temperature bias ΔT generates a ...

Table 5: Examples of features with high input scores in early layers. Activated token are marked with an underline.

"Findings show that"	"It's no surprise that"	"It's been a long time since"
"I once heard that"	"Have you ever noticed that"	"In my experience,"
"Then the man said:"	"I couldn't believe when"	"The craziest part was when"
"I believe that"	"The first thing I heard was"	"If you think about it,"
"The news mentioned"	"Let me tell you a story about"	"I was shocked to learn that"
"She saw a"	"Someone once told me that"	"For some reason,"
"It is observed that"	"It might sound strange, but"	"I can't help but wonder if"
"Studies indicate that"	"They always warned me that"	"It makes sense that"
"According to reports,"	"Nobody expected that"	"At first, I didn't believe that"
"Research suggests that"	"Funny thing is,"	"That reminds me of the time when"
"It has been noted that"	"I never thought I'd say this, but"	"It all comes down to"
"I remember when"	"What surprised me most was"	"One time, I saw that"
"It all started when"	"The other day, I overheard that"	"I was just thinking about how"
"The legend goes that"	"Back in the day,"	"Imagine a world where"
"If I recall correctly,"	"You won't believe what happened when"	"They never expected that"
"People often say that"	"A friend of mine once said,"	"I always knew that"
"Once upon a time,"	"I just found out that"	

Table 6: Neutral prefixes used for generation for main steering experiments.

by Wu et al. (2025). We provide additional details on each of these methods:

- **Prompt.** Given a concept, an external LLM generates a steering prompt. This prompt is then pre-pended to the actual instructions.
- **LoRA.** A parameter-efficient finetuning method (Xu et al., 2024), trained to minimize the language modeling loss on a set of positive examples that contain the concept.
- **LoReFT.** Low-rank representation finetuning (Wu et al., 2024), with the same objective as LoRA.
- **ReFT-r1.** A weakly supervised approach based on Wu et al. (2024) (Wu et al., 2025). ReFT-r1 jointly learns concept detection and steering on supervised data by combining the training objectives of linear probing and supervised steering.
- **Difference-in-means.** Difference-in-means (DiffMean) (Larsen et al., 2016) uses the difference between averaged representations from two classes of inputs as a steering vector.
- **SAE-A.** SAEs with AUROC selection (Wu et al., 2025). For each feature, its max-pooled activations are computed over a set of positive training examples. Top-scoring features, based on the AUROC over the dataset given true labels, are used for steering.

License. AxBench and Alpaca-Eval are published under an Apache license. GSM8K is published under MIT license. Dolly is published under Creative Commons Attribution Share Alike 3.0 license. All artifacts are used in compliance with their license.

G Computational Requirements

The experiments in this work were conducted using an NVIDIA A40 node with 8 48GB GPUs. The complete set of experiments, including initial exploration, consumed 300-500 GPU hours.

Computing the output score for a single feature takes 6 seconds on average on a single A40 48GB GPU. Computing the input scores given precomputed activations takes negligible time.

Concept Score

[System]
Please act as an impartial judge and evaluate whether the specified concept is presented in the sentence fragment provided below. Focus solely on whether the concept is clearly incorporated, without regard for grammar or logical coherence. Begin your evaluation by providing a short explanation of whether the concept is clearly incorporated. Point out references to the concept in the fragment, noting any minimal or awkward incorporations.
After providing your explanation, rate the concept's relevance on a scale from 0 to 2, where 0 indicates the concept is not present at all, 1 indicates the concept is somewhat present but minimally or awkwardly incorporated, and 2 indicates the concept is more fully and effectively incorporated, with stronger and more natural integration.
Provide your rating using this exact format: "Rating: [[score]]".

[Concept Start]
[Concept goes here]
[Concept End]
[Sentence Fragment Start]
[Sentence goes here]
[Sentence Fragment End]

Table 7: Prompt used to compute the concept score.

Instruct Score

[System]
Please act as an impartial judge and evaluate whether the sentence fragment provided below is related to the instruction. Focus solely on the degree of relatedness in terms of topic, regardless of grammar, coherence, or informativeness. Begin your evaluation by providing a brief explanation of whether the sentence is related to the instruction, and point out references related to the instruction. After providing your explanation, rate the instruction relevance on a scale from 0 to 2, where 0 indicates the sentence is unrelated to the instruction, 1 indicates it is somewhat related but only minimally or indirectly relevant in terms of topic, and 2 indicates it is more clearly and directly related to the instruction.
Provide your rating using this exact format: "Rating: [[score]]".

[Instruction Start]
[Instruction goes here]
[Instruction End]
[Sentence Fragment Start]
[Sentence goes here]
[Sentence Fragment End]

Table 8: Prompt used to compute the instruct score.

Fluency Score
[System] Please act as an impartial judge and evaluate the fluency of the sentence fragment provided below. Focus solely on fluency, disregarding its completeness, relevance, coherence with any broader context, or informativeness. Begin your evaluation by briefly describing the fluency of the sentence, noting any unnatural phrasing, awkward transitions, grammatical errors, or repetitive structures that may hinder readability. After providing your explanation, rate the sentence's fluency on a scale from 0 to 2, where 0 indicates the sentence is not fluent and highly unnatural (e.g., incomprehensible or repetitive), 1 indicates it is somewhat fluent but contains noticeable errors or awkward phrasing, and 2 indicates the sentence is fluent and almost perfect. Provide your rating using this exact format: "Rating: [[score]]". [Sentence Fragment Start] [Sentence goes here] [Sentence Fragment End]

Table 9: Prompt used to compute the fluency score.