Training Language Models to Explain Their Own Computations

Belinda Z. Li [1 2] **Zifan Carl Guo** [2] **Vincent Huang** [1] **Jacob Steinhardt** [1] **Jacob Andreas** [1 2]

Abstract

Can language models (LMs) learn to faithfully describe their internal computations? Are they better able to describe themselves than other models? We study the extent to which LMs' privileged access to their own internals can be leveraged to produce new techniques for explaining their behavior. Using existing interpretability techniques as a source of ground truth, we fine-tune LMs to generate natural language descriptions of (1) the information encoded by LM features, (2) the causal structure of LMs' internal activations, and (3) the influence of specific input tokens on LM outputs. When trained with only tens of thousands of example explanations, explainer models exhibit non-trivial generalization to new queries. This generalization appears partly attributable to explainer models' privileged access to their own internals: fine-tuning a model to explain its *own* computations generally works better than fine-tuning a different model (even if the explainer model is significantly more capable than the target). Our results suggest not only that LMs can learn to reliably explain their internal computations, but that such explanations offer a scalable complement to existing interpretability methods.[1]

1. Introduction

When language models (LMs) are prompted to explain decisions, their outputs are often plausible (Korbak et al., 2025), but may not bear any relationship to the computation that first gave rise to LMs' decisions. Indeed, work using attribution, feature visualization, and mechanistic interpretability techniques (Simonyan et al., 2013; Covert et al., 2020; Templeton et al., 2024) has shown that LMs' verbalized explanations sometimes systematically fail to describe the factors that determine their decisions (Turpin et al., 2023; Chen et al., 2025; Barez et al., 2025).

Can we fix this problem by training LMs so that their verbalized explanations faithfully describe their internal computations? Several recent papers (Binder et al., 2025; Song et al., 2025b; Plunkett et al., 2025) have studied the related question of whether models can describe features of their own *output distributions*, arguing that models might be able to leverage **privileged access** to their own internals to do so effectively. In this paper, we are interested in an even deeper form of privileged access: whether models can learn to describe not only their outputs, but their *internal* representations and mechanisms, and whether privileged self-access enables them to do so better than learned explanation-generating methods derived from other models. We state our main hypothesis as follows:

The Privileged Access Hypothesis
Models trained to explain their own internal computations can do so more accurately than other models trained to explain them.

To explore this hypothesis, we fine-tune various *explainer* models to predict three aspects of *target* models' internal procedures:[2]

1. Descriptions of internal features, specifically what inputs activate them (**feature descriptions**; Figure 1A).
2. Outcomes of interventions to internal activations (**activation patching**; Figure 1B).
3. Descriptions of decision rules, specifically the important tokens for a decision (**input ablation**; Figure 1C). This setting bears similarities to Binder et al. (2025) due to measuring self-access to the model's output distributions.

[1]Transluce [2]MIT CSAIL. Correspondence to: Belinda Z. Li <bzl@mit.edu>.

Preprint. February 10, 2026.

[1]Code available at `https://github.com/TransluceAI/introspective-interp`.

[2]This paper treats the *target model* as fixed and only fine-tunes the *explainer model*. While this setup does not constitute *self*-explanation in the strictest sense (since the explainer's output distribution changes), we believe it is more useful to explain pre-trained models than enforce this strict notion of self-consistency. We discuss the implications of this simplification in Section 7. For the rest of this paper, we use the term "self-explaining" in this looser sense.

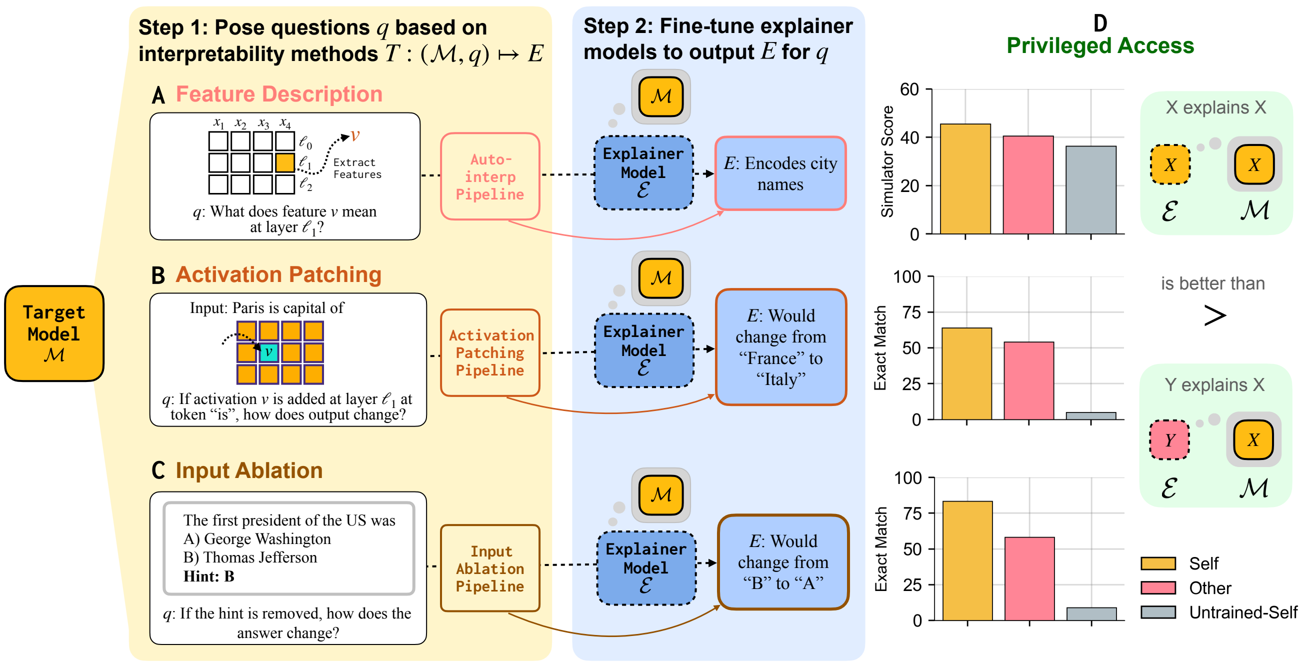
Step 1: Pose questions q based on interpretability methods $T : (\mathcal{M}, q) \mapsto E$
A Feature Description
Extract Features
q: What does feature v mean at layer ℓ_1?
B Activation Patching
Input: Paris is capital of
q: If activation v is added at layer ℓ_1 at token "is", how does output change?
C Input Ablation
The first president of the US was A) George Washington B) Thomas Jefferson Hint: B
q: If the hint is removed, how does the answer change?
Target Model $\mathcal{M}$
Auto-interp Pipeline
Activation Patching Pipeline
Input Ablation Pipeline
Step 2: Fine-tune explainer models to output E for q
Explainer Model $\mathcal{E}$
E: Encodes city names
E: Would change from "France" to "Italy"
E: Would change from "B" to "A"
D Privileged Access
Simulator Score
Exact Match
X explains X
is better than
Y explains X
Self
Other
Untrained-Self

Figure 1. Overview of our methods for training models to describe their own internal procedures. As a first step, we extract answers to three types of question about the target model using different interpretability methods: (A) descriptions of features of the target model's hidden representations via an auto-interp procedure, (B) explanations about what parts of the model affect the output via activation patching, (C) explanations about what parts of the input affect the output via input ablation. As a second step, we fine-tune three explainers to produce each of the three types of explanations. (D) After fine-tuning various explainer models on various target models, we find evidence of privileged access for all three explanation types, and use this to obtain data-efficient explainers.

Our experiments find evidence of privileged access—the target model is better explained by itself than by other explainer models, even when the other model is more capable (e.g. larger or instruction-tuned). Diving deeper, we find:

1. **LMs can be fine-tuned to self-explain**: As in past work (Li et al., 2025), we find that explanation capabilities are not always natively present in models, and must be acquired via fine-tuning. A small amount of fine-tuning enables LMs to faithfully describe their internal features, significantly outperforming zero-shot methods (Kharlapenko et al., 2024; Chen et al., 2024a) and nearest-neighbor SAE features (Huben et al., 2024). Furthermore, a model's ability to explain (its own or others') features is correlated with the degree of similarity between the explainer and target models (§3.4).
2. **Self-explaining is data-efficient**: Self-explaining is approximately a hundred times more sample-efficient than a nearest neighbors baseline, achieving comparable results with only 0.8% of the training data (§4).
3. **Privileged access extends across tasks**: we find patterns of privileged access in all three domains, including activation patching (§5.1) and input ablation (§5.2).

These results suggest that even when off-the-shelf LMs cannot faithfully self-interpret, they can learn to do so through an objective that enforces consistency between their output and their internal procedures. Our approach reframes interpretability as not only an external analysis problem, but as a capability that can be trained into LMs themeselves; by leveraging privileged access to internal computations, "introspective interpretability" techniques offer an avenue towards scalable understanding of LM behavior.

2. Methods

In this section we describe a general framework for training models to verbalize different aspects of their internal states, and then instantiate it with three specific explanation types.

2.1. Definitions and Task

Abstractly, a neural **model** $\mathcal{M}$ can be represented as a pipeline $x \to h \to y$ that maps **inputs** x to **hidden representations** h to **outputs** y. An **explanation** E of the neural network may enable us to answer **questions** q about what information in x is represented in h and how that information influences y. For example, we may wish to answer a question $q =$ *What inputs activate direction* v *at layer* l*?* If v activates on tokens *Tokyo*, *NYC*, and *Paris*, then an appropriate explanation E might be *city names*. We are interested

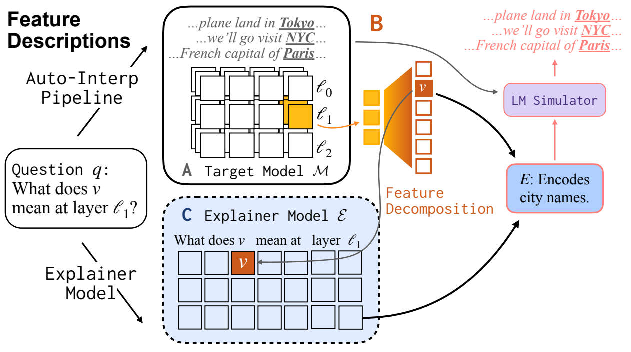
Feature Descriptions
Auto-Interp Pipeline
Question q: What does v mean at layer ℓ_1?
Explainer Model
...plane land in Tokyo...
...we'll go visit NYC...
...French capital of Paris...
ℓ_0
ℓ_1
ℓ_2
A Target Model $\mathcal{M}$
B
v
Feature Decomposition
LM Simulator
E: Encodes city names.
C Explainer Model $\mathcal{E}$
What does v mean at layer ℓ_1

Figure 2. Training an explainer to predict **feature descriptions**. (A) Run the target model $\mathcal{M}$ on many inputs. (B) For each vector v at layer ℓ, pick an explanation E that best matches the context where v is active, as judged by a LM simulator. (C) Train the explainer to output E given the question q about v at layer ℓ. The trained explainer generalizes well to both held-out and OOD v.

in whether LMs can learn to generate E directly.

In this paper, we will train **explainer models** $\mathcal{E}$ to produce explanations E for various **target models** $\mathcal{M}$ and types of questions q. Abstractly, let $T : (\mathcal{M}, q) \mapsto E$ be a (model-external) **explanation procedure** that generates an explanation E answering a question q about model $\mathcal{M}$. We use T as supervision to train an explainer model, by minimizing the cross-entropy loss on the explanation:

$$\mathcal{L}_{\mathcal{E}} = -\mathbb{E}_q \left[\log p_{\mathcal{E}}(E = T(\mathcal{M}, q) \mid q)\right]. \qquad (1)$$

Our experiments focus on explaining Transformer language models $\mathcal{M}$ (Grattafiori et al., 2024; Yang et al., 2025), which consist of layers ℓ each containing attention and MLP blocks, and *residual stream* representations h_ℓ in between each layer. We also use Transformer language models as explainers $\mathcal{E}$. Throughout this paper, we use notation $x = (x_1, \cdots x_n)$ to refer to an input sequence with token positions $t \in [n]$. We use $h_{(\ell,t)}(x)$ to denote the residual stream activation of $\mathcal{M}$ at layer ℓ and token position t for input x.

We will train explainer LMs to answer three types of questions q, described below.

2.2. Feature Descriptions: What does a model-internal feature represent?

Question q. For feature descriptions, our question q takes form *what types of inputs activate direction v in the residual stream at layer l?*

Interpretability procedure T. We make use of automated interpretability methods (Hernandez et al., 2022; Bills et al., 2023), which generate natural-language descriptions E of input tokens on which a given v is highly active.

Formally, let $a_v(x, \ell, t) = \langle h_{\ell,t}(x), v\rangle$ denote the "activation" of v in the residual stream representation of token x_t (i.e. the alignment between $h_{\ell,t}$ and v). To obtain feature descriptions, we use a "simulator" language model to simulate how v would activate on an input sequence x if it had description E, outputting an "expected" activation pattern $\hat{a}(x, t, E)$. See Figure 2B. We train the simulator following Choi et al. (2024) using Neuronpedia SAE labels and FineWeb exemplars (Penedo et al., 2024).

We then search for the explanation E that maximizes correlation between simulated and true activations over inputs:

$$T_{\text{feat}}(\mathcal{M}, v, \ell) = \arg\max_{E} \mathbb{E}_x \left[\texttt{corr}_t \left(a_v(x, \ell, t),\ \hat{a}(x, t, E)\right)\right] \qquad (2)$$

The prompts to encode q and E are given in Appendix F.

Training the explainer. We take v from the layer-ℓ residual stream of the target model and pass it as a continuous token to the explainer by inserting v at the embedding layer of the explainer model.[3] See Figure 2C.

For explainer models whose hidden dimension do not match the target model's, we introduce a **linear projection** $\Pi_\ell \in \mathbb{R}^{d_{\mathcal{E}} \times d_{\mathcal{M}}}$ from the target model hidden size $d_{\mathcal{M}}$ to the explainer model hidden size $d_{\mathcal{E}}$, which is trained jointly with the rest of LM parameters. We learn a separate projection per layer ℓ of the target model. We train explainers and projections to optimize:

$$\boxed{\mathcal{L}_{\text{feat}} = -\mathbb{E}_{v,\ell}\left[\log p_{\mathcal{E}}(E = T_{\text{feat}}(\mathcal{M}, v, \ell) \mid \Pi_\ell\, v, \ell)\right]} \qquad (3)$$

Choice of directions v. Our experiments train on Sparse Auto-Encoder (SAEs) features obtained from sparse dictionary learning approaches Huben et al., 2024; Bricken et al., 2023), but test generalization to full activations and activation differences. See details in Section 3.2.

2.3. Activation Patching: What internal components affect the prediction?

Question q. We ask which hidden components are *causally important* for a model's output (Meng et al., 2022; Zhang & Nanda, 2024). Specifically, the question q is *on input x, (how) does changing the activation at layer ℓ and token x_t affect the output?* An example is shown in Figure 3.

Interpretability Procedure T. We can answer questions of this form through activation patching and circuit tracing methods. Given an input x (*Paris is the capital of* in Figure 3), an activation patching experiment typically constructs an alternative input x' (*Rome is the capital of*). We

[3] In early experiments, we also investigated explaining features v from other components of the target model, but found that explainers struggled to reproduce these explanations. We hypothesize that we need to patch features into the same type of location (i.e. the embedding layer is part of the residual stream).

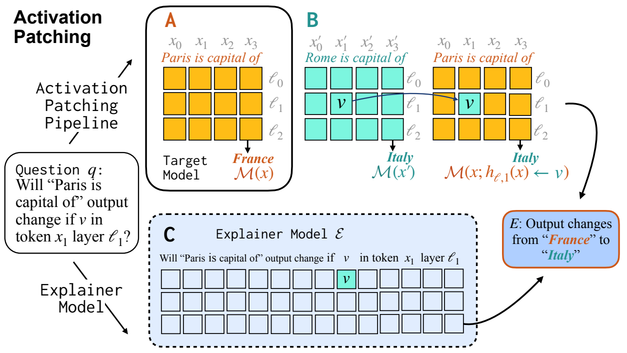
Activation Patching
Activation Patching Pipeline
Question q: Will "Paris is capital of" output change if v in token x_1 layer ℓ_1?
Explainer Model
A
Paris is capital of
Target Model
France
$\mathcal{M}(x)$
B
Rome is capital of
Italy
$\mathcal{M}(x')$
Paris is capital of
Italy
$\mathcal{M}(x; h_{\ell,1}(x) \leftarrow v)$
C Explainer Model $\mathcal{E}$
Will "Paris is capital of" output change if v in token x_1 layer ℓ_1
E: Output changes from "France" to "Italy"

Figure 3. Training an explainer to predict **activation patching outcomes**. (A) Run target model $\mathcal{M}$ on input x to obtain $\mathcal{M}(x)$. (B) Perform activation patching by running $\mathcal{M}$ on a counterfactual input x'—in this example, patching in vector v from token x_1 and layer ℓ_1 of the counterfactual run into the original run—and then construct an explanation E about how the resulting prediction changes. (C) Train an explainer model to answer E when given a question q about the patching procedures.

then run a forward pass of $\mathcal{M}$ on x, but replace the activation $h_{\ell,t}(x)$ with the corresponding activation $h_{\ell,t}(x')$:

$$\mathcal{M}\left(x;\, h_{\ell,t}(x) \leftarrow h_{\ell,t}(x')\right)\ , \tag{4}$$

and we measure the change in the model's output:

$$\begin{aligned} T_{\text{patch}}(\mathcal{M}, q) &= T_{\text{patch}}(\mathcal{M}, x, \ell, t) \\ &= d\Big(\mathcal{M}(x),\ \mathcal{M}\big(x;\, h_{\ell,t}(x) \leftarrow h_{\ell,t}(x')\big)\Big). \end{aligned} \tag{5}$$

where $d(\cdot,\cdot)$ is some measure of the difference between the two predictions. This process is illustrated in Figure 3B.[4]

Training the explainer. Rather than directly predicting the numerical score in Equation (5), we train explainers $\mathcal{E}$ to predict (1) *whether* the model output changes under patching, and (2) the *content* of the model output after patching. Similarly to Section 2.2, we insert v as a continuous token in the embedding layer of the explainer model. See Figure 3C.

Finally, we optimize:

$$q \triangleq (x, t, \ell_{1\cdots i}, v). \tag{6}$$

$$\boxed{\mathcal{L}_{\text{patch}} = -\mathbb{E}_{x,x',q}\Big[\log p_{\mathcal{E}}\big(E = T_{\text{patch}}(\mathcal{M}, q) \mid q\big)\Big]} \tag{7}$$

The exact prompt and output templates are in Appendix F.

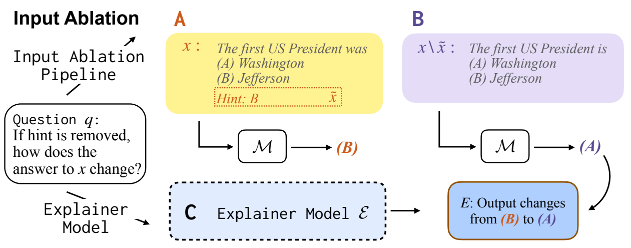
Input Ablation
Input Ablation Pipeline
Question q: If hint is removed, how does the answer to x change?
Explainer Model
A
x : The first US President was (A) Washington (B) Jefferson Hint: B $\tilde{x}$
$\mathcal{M}$
(B)
B
$x \backslash \tilde{x}$: The first US President is (A) Washington (B) Jefferson
$\mathcal{M}$
(A)
C Explainer Model $\mathcal{E}$
E: Output changes from (B) to (A)

Figure 4. Training an explainer to predict **input ablation outcomes**. (A) Run a target model $\mathcal{M}$ on an input x. (B) Identify whether input component $\tilde{x} \subset x$ affected its output by rerunning $\mathcal{M}$ on $x \backslash \tilde{x}$. (C) Train an explainer model to answer question q about how the output changes when $\tilde{x}$ is removed from x.

2.4. Input Ablation: What parts of the input matter?

Question q. For input ablation, we ask which input tokens most influence the model's prediction (Chang et al., 2025; Vafa et al., 2021), which we can identify by ablating subsets $\tilde{x}$ of the input x. Specifically, q takes form *if $\tilde{x}$ is removed, (how) does the answer to x change?* See Figure 4.

Interpretability Procedure T. Input ablation measures how much the target model $\mathcal{M}$'s output changes when we take an input x (e.g. *The first US president...* in Figure 4) and remove a subset of tokens $\tilde{x} \subseteq x$ (e.g. *Hint: B*):

$$T_{\text{input}}(\mathcal{M}, x, \tilde{x}) = d\left(\mathcal{M}(x),\ \mathcal{M}(x \backslash \tilde{x})\right), \tag{8}$$

where d measures difference in the model's outputs. This process is illustrated in Figure 4B.

Training the explainer. $\mathcal{E}$ is asked to predict (1) *whether* removing $\tilde{x}$ would change the $\mathcal{M}$'s answer, and (2) the *content* of $\mathcal{M}$'s new output without $\tilde{x}$. See Figure 4C. Formally, the objective we optimize is:

$$\boxed{\mathcal{L}_{\text{input}} = -\mathbb{E}_{x,\tilde{x}}\left[\log p_{\mathcal{E}}(E = T_{\text{input}}(\mathcal{M}, x, \tilde{x}) \mid x, \tilde{x})\right]} \tag{9}$$

Full prompt and output templates are in Appendix F.

3. Privileged Access Improves Explanations

In this section, we show that (1) models can be fine-tuned to generate feature descriptions and generalize to new features and feature bases, and (2) this capability relies on privileged access. We focus on feature description (§2.2), and study additional tasks (patching, input ablation) in §5.

[4]To ensure sufficient samples where the model's output changes We divide the target model's layers into four blocks, and perform activation patching over one block of layers at a time We aggregate the representations over the block $v = \text{avg}_{\ell_k}\left(h_{\ell_k,t}(x')\right)$ before inserting into all corresponding layers of the original input, allowing us to specify only a single continuous token to the explainer.

3.1. Models and Training Data

Models. We use Llama-3.1-8B (Grattafiori et al., 2024) as the target model,[5] and train five different explainer models (Llama-3.1-8B, Llama-3-8B, Llama-3.1-8B-Instruct, Llama-3.1-70B, Qwen3-8B; Yang et al., 2025).[6] For Llama-3.1-8B, we report results with both LoRA (Hu et al., 2022) and full-parameter fine-tuning. For Llama-3.1-70B we only report LoRA; for other models we only report full fine-tuning.

Initializing Π_ℓ. For models that require projection layers Π_ℓ due to hidden-size mismatch (Qwen3-8B, Llama-3.1-70B), we *randomly initialize* Π_ℓ and train them with the rest of the parameters.[7] For Llama-3.1-70B, we also include results where Π_ℓ was *pre-trained* on $\min_{\Pi_\ell} \|h^{\mathcal{E}}_{\ell,t} - \Pi_\ell \cdot h^{\mathcal{M}}_{\ell,t}\|_2$ (distance between explainer and projected target activations) on FineWeb samples (Penedo et al., 2024),

Data. We train on residual stream SAE features of Llama-3.1-8B target models from Llama-Scope (He et al., 2024). We use the Llama-3.1-8B-LXR-32x features, which maps the residual stream after each layer to 131K features (32 times larger than the residual hidden dimension). Ground-truth explanations E for each feature were obtained from Neuronpedia (Lin, 2023).

3.2. Evaluation Set and Metrics

We train on a subset of SAE features, then evaluate generalization in- and out-of-distribution on three types of features:

Held-out SAE features (SAE): We hold out a subset of 1550 SAE features (50 per layer) for testing.

Full Activations (ACT): We test out-of-distribution generalization to full residual stream activations $h_{(\ell,t)}(x)$ on inputs x sampled from FineWeb (Penedo et al., 2024).

Activation Differences (ΔACT): Differences between counterfactual pairs of inputs allow us to isolate the effect of a single pointwise change. We create counterfactual pairs (x, x') from the Counterfact dataset (Meng et al., 2022) using the procedure described in Section 2.3 and extract activation differences $v = h_{\ell,t}(x) - h_{\ell,t}(x')$.

To measure explainer performance on each type of feature, we use two metrics. **(1)** For SAE features, gold descriptions exist, so an LM judge can assess the similarity of predictions to the gold description, on a scale of 0 to 1 in increments of 0.25. The LM judge has 81.25% human agreement pe Appendix A. **(2)** We also use *simulator score* (Equation (2)), measuring the Pearson correlation between true and predicted activations based on $\mathcal{E}(v)$ for each sample x.

	SAE			
Explainer	LM Judge	Sim.	ACT	ΔACT
Llama-3.1-8B	$\mathbf{76.2}_{\pm 0.9}$	$\mathbf{45.1}_{\pm 0.7}$	$\mathbf{49.7}_{\pm 0.6}$	$32.0_{\pm 0.8}$
Llama-3.1-8B (LoRA)	$\mathbf{76.0}_{\pm 0.9}$	$\mathbf{45.6}_{\pm 0.8}$	$\mathbf{50.6}_{\pm 0.6}$	$\mathbf{34.0}_{\pm 0.8}$
Llama-3-8B	$\mathbf{77.0}_{\pm 0.9}$	$\mathbf{44.6}_{\pm 0.7}$	$49.3_{\pm 0.6}$	$32.4_{\pm 0.7}$
Llama-3.1-8B-Inst.	$\mathbf{77.1}_{\pm 0.8}$	$42.7_{\pm 0.7}$	$46.9_{\pm 0.7}$	$29.9_{\pm 0.7}$
Qwen3-8B	$70.3_{\pm 0.9}$	$40.6_{\pm 0.8}$	$21.1_{\pm 0.7}$	$12.3_{0.4}$
Llama-3.1-70B (LoRA)				
random proj.	$63.9_{\pm 1.0}$	$39.5_{\pm 0.8}$	$12.6_{\pm 0.4}$	$12.2_{\pm 0.4}$
pre-trained proj.	$74.1_{\pm 0.9}$	$\mathbf{45.2}_{\pm 0.7}$	$33.8_{\pm 0.7}$	$20.6_{\pm 0.6}$
Nearest Neighbors	$58.5_{\pm 1.0}$	$33.7_{\pm 0.8}$	$38.9_{\pm 0.7}$	$18.5_{\pm 0.6}$
SelfIE Best of 5	$40.1_{\pm 1.0}$	$36.4_{\pm 0.8}$	$43.3_{\pm 0.6}$	$21.0_{\pm 0.6}$
Gold SAE Labels	$100_{\pm 0.0}$	$43.3_{\pm 0.8}$	-	-

Table 1. **A target model's features are best explained by itself and its variants.** Comparison of explainer methods on Llama-3.1-8B residual SAE features, including trained models, nearest neighbor, SelFIE, and gold SAE label baselines. We report LM judge and simulator scores (mean $\pm$ SE) for held-out SAE features, and simulator scores for out-of-domain activations (ACT) and differences (ΔACT), all scaled to 100. **Bold** means not significantly different from best (paired t-test, $p \geq 0.05$). Our methods can even beat gold SAE labels as scored by a simulator. **Activation alignment improves explainer performance**: pre-training a projection to align features improves Llama-3.1-70B's performance.

3.3. Baselines

Nearest Neighbors. To evaluate whether explainer models exhibit nontrivial generalization to unseen features, we compare to a nearest-neighbor baseline that, given a new feature or representation, finds the single most similar feature from the training set, and outputs its associated description:

$$\mathcal{E}_{\text{NN}}(v) = D\left(\underset{v_i \in S_{\text{train}}}{\arg\max} \langle v_i, v \rangle\right).$$

where S_{train} denotes the set of features used for training. This gives us an estimate of roughly how well the explainer can do if it simply learned to memorize and interpolate samples from the training data.[8]

SelfIE. Following SelfIE (Chen et al., 2024a), we directly patch v into the embedding layer of a non-finetuned Llama-3.1-8B model,[9] and prompt it to define the feature. We use standard prompts found to work well for self-explanations (Kharlapenko et al., 2024).[10] Because SelfIE is sensitive to the scale of v when it is inserted (Kharlapenko et al., 2024), we report results on the *best* explanation across

[5]We run additional experiments with Gemma 2 (Gemma Team et al., 2024) as target model, see Section C.1.

[6]Details can be found in Section B.1.

[7]For LoRA training, these layers are included in the set of parameters trained via low-rank updates.

[8]We also perform *layer-wise* nearest neighbors in Section B.2.

[9]The original SelfIE paper patches representations into the third layer, but prior work has found that patching into the first few layers work about equivalently well (which we also find). Thus, we patch every activation into the embedding layer.

[10]SelfIE prompts can be found in Section H.1.

5 possible scales (v, $5v$, $10v$, $25v$, and $50v$), representing an upper bound on SelfIE performance.

Gold SAE labels. For held-out SAE features, we evaluate their **gold Neuronpedia labels** with the simulator.

3.4. Results

Results can be found in Table 1. We observe:

Trained explainers are effective. For every feature type and metric, training Llama-3.1-8B to explain itself outperforms all baselines by a significant margin. This even includes gold SAE labels due to noise in the ground-truth feature labels themselves (see error analysis in Section A.3).

Alignment between activations improves verbalization capability. Among all trained models, the Llama-3.1-8B and Llama-3-8B models consistently outperform Llama-3.1-8B-Instruct by a few points and significantly outperform Qwen3-8B and Llama-3.1-70B, despite Llama-3.1-70B's larger size. We posit that activation similarity between the explainer and target model predicts explainer performance.

We confirm this correlation visually by plotting explainer-target activation similarity vs. explainer performance in Section C.2. the explanation capabilities of Llama-3.1-70b with the *pre-trained* projection and *randomly initialized* projection serve as a proxy for maximally and minimally aligned activation, respectively, given the model. As reported in Table 1, we can recover a significant fraction of performance when we start training from the pre-trained projection, performing 14% better on SAE explanations, 2.7 times better on real activations, and 1.7 times better on activation differences. We conclude that:

For feature descriptions, privileged access improves model explanations: models are better explained by fine-tuned versions of themselves than by other models.

4. Privileged Access Confers Data-Efficiency

To answer where and why privileged access may be useful, we investigate whether it confers data-efficiency advantages over alternatives. We train Llama-3.1-8B, Qwen3-8B, and the nearest neighbor baseline on subsets of the training data of various sizes and plot scaling curves against SelfIE baseline in Figure 5.

In general, self-explanation is more data-efficient than training another explainer model or using nearest neighbors. At just 0.8% of all training features (1024 samples per layer), Llama reaches 71%, 80%, 89%, and 81% of its end-performance on two metrics for held-out SAEs and simulator score for ACT and ΔACT respectively; Qwen reaches 35%, 24%, 46%, and 66%, and nearest neighbors reaches 55%, 56%, 55%, and 75%. As Llama's end-performance is already higher than Qwen and nearest neighbors, this gap only widens in low-data regimes. Thus,

Training a model to generate descriptions of its own features is data-efficient, particularly compared to training other models to describe the target model's features.

This data-efficiency underscores the practical utility of self-explanation: annotating feature descriptions is expensive, and self-explanation training allows us to recover most of the performance of other explanation techniques with orders of magnitude less data.

5. Generalization Across Tasks

We check if self-explanation and privileged access generalize to other domains by training models to explain outcomes of activation patching (§2.3) and input ablation (§2.4).

Models. Across both tasks, we train two explainer models, Qwen3-8B and Llama-3.1-8B, to explain Qwen3-8B. We also report results with Llama-3.1-8B as the target model for activation patching and input ablations in Sections D.1 and E.1 respectively.

Metrics. Recall from Sections 2.3 and 2.4 that the explainer output for both activation patching and input ablations typically includes two parts: whether the target output would changed under intervention (*changes / remains unchanged*), and the content of the change (*to France*). Based on this, we report three different types of metrics:

1. Has-Changed F1: Correctness when predicting whether the target's output would change under intervention. We report the Macro F1 scores over the "changed" and "unchanged" classes.

2. Content Match: Correctness of predictions about the *content* of target model output under intervention.

3. Exact Match: Exact match accuracy between generated explanation and ground-truth explanation. Requires the explainer to correctly predict both parts.

Baselines. To measure the effect of fine-tuning, we prompt Qwen3-8B (without any explanation training) to generate explanations of its own activation patching and input ablation procedures. The full prompts for untrained activation patching are in Section H.2. and for untrained input ablations are in Section H.3. These baselines are analogous to the SelfIE baseline in Section 3.3.

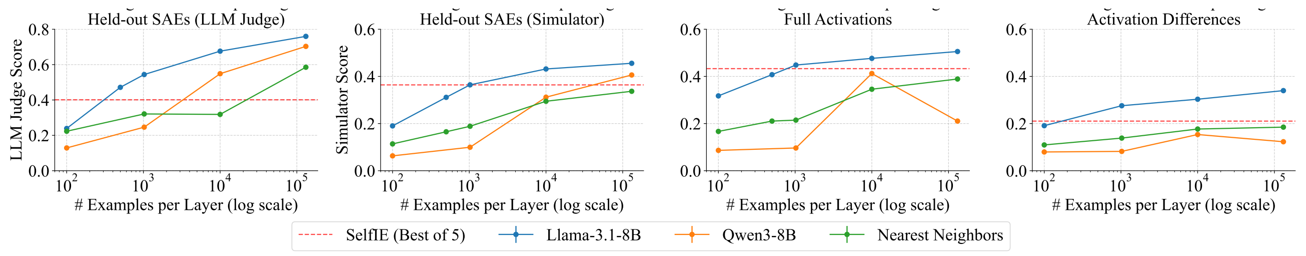

Figure 5. **Training self-as-explainer is more data-efficient than alternatives.** Scaling curves of training samples per layer vs. explanation quality (as measured either by LM judge or simulator) on three types of features (SAE, full activations, activation differences) show that matching explainer to target (Llama-3.1-8B) is more data-efficient than using a different model (Qwen) or nearest neighbors, especially in the low-data regime. The latter requires more samples per layer to outperform an untrained SelfIE baseline. Note that we aggregate 5 attempts of SelfIE with different strength, so the graph shows an upper bound of SelfIE performance.

		Activation Patching			Input Ablation		
Target	Explainer	Exact Match	Has-Changed F1	Content Match	Exact Match	Has-Changed F1	Content Match
Qwen3-8B	Qwen3-8B	**64.0**$_{\pm 0.4}$	**80.2**$_{\pm 0.3}$	**71.0**$_{\pm 0.4}$	**83.4**$_{\pm 1.0}$	**87.0**$_{\pm 1.1}$	**90.6**$_{\pm 0.8}$
	Llama-3.1-8B	54.1$_{\pm 0.4}$	73.9$_{\pm 0.4}$	65.4$_{\pm 0.4}$	58.1$_{\pm 1.3}$	70.5$_{\pm 0.0}$	71.6$_{\pm 1.2}$
	Qwen3-8B (Untrained)	5.02$_{\pm 0.2}$	24.4$_{\pm 0.8}$	18.8$_{\pm 0.3}$	8.9$_{\pm 1.5}$	44.4$_{\pm 1.6}$	35.3$_{\pm 2.5}$

Table 2. **A target model's intervention outcomes are best predicted by itself, shown through activation patching and input ablation.** We train Qwen3-8B to verbalize its own intervention outcomes, and compare against training a different model (Llama-3.1-8B), and also against an untrained version of Qwen3-8B. Scores (mean ± standard error) are reported for three metrics—exact match, has-changed F1, and content prediction. For each task and metric, **bold** indicates no significant difference from the best entry (paired t-test, $p \geq 0.05$). Results with Llama-3.1-8B as the target model can be found in Tables 5 and 6.

5.1. Activation Patching: Training LMs to Predict the Outcomes of Activation Interventions

We train models to explain outcomes of activation patching, which perturbs a model component and examines whether it changes the model output, described in detail in Section 2.3.

Training Data. We use the CounterFact dataset (Meng et al., 2022) as a source of counterfactual input pairs x, x', which consists of a corpus of factual subject–relation–object sentences like *Paris (subject) is the capital of (relation) France (object)*. We prompt the model with the subject and relation and ask the model to predict the object . We then identify a counterfactual sentence x' with the same relation but differing subject and object (*Rome is the capital of*) and measure whether the most likely LM prediction changes (e.g. to *Italy*). We perform sampling to ensure an even split between changed and unchanged predictions, at each token position and layer chunk. See Section G.2 for details.

Results for activation patching can be found in Table 2. The privileged access hypothesis is supported: Qwen3-8B is best at explaining Qwen3-8B. All trained explainers perform better than their untrained versions, which generally guess the has-changed value randomly. We also include a set of ablation experiments in Section D.2 to isolate which input component(s) the explainer learned to condition on.

5.2. Input Ablations: Training LMs to Describe their Decision Rules

Using the methods described in Section 2.4, we train models to explain how withholding or including parts of target models' *input* would affect the output prediction.

Training Data. Following recent work used to study faithful chain-of-thought (Chen et al., 2025), we give $\mathcal{M}$ inputs in the form of a multiple-choice question c with an injected hint $\tilde{x}$, i.e. $x = [c, \tilde{x}]$, and train the explainer to predict whether $\mathcal{M}$'s most likely next token will change to $\tilde{x}$. Previously, Chen et al. (2025) found that models fail to verbalize $\tilde{x}$ in their chain-of-thought, despite $\tilde{x}$ being critical to its prediction ($\mathcal{M}([c, \tilde{x}]) \neq \mathcal{M}(c)$). Thus, we investigate whether models can be trained to detect *when and how* their prediction will change depending on $\tilde{x}$'s presence.

We use MMLU (Hendrycks et al., 2021) as the source of multiple-choice reasoning question, and inject hints of the form $\tilde{x} =$ "*Hint: A*" to the end of the question. We randomly select $\tilde{x}$ as one of the four multiple-choice options. To make the has-changed prediction nontrivial, we construct the hint prompt such that we have a roughly equal number of samples that change and do not change according to the hint. See Section G.3 for details.

Results for input ablation are also shown in Table 2. We find similar evidence for *privileged access*, where Qwen3-

8B is best explained by itself than other models. Finally, we examine untrained Qwen's predictions and find that it a has-changed value of `True` only 8.6% of the time, and most of its content match comes from correctly predicting the unchanged answers (96.4% content match) compared to its changed answers (10.3% content match). This is consistent with findings on reasoning models in prior work: Chen et al. (2025) finds that no pre-trained LM would ever report using a hint. This indicates that explicit fine-tuning is essential to elicit faithful explanations of decision rules. Taken together, results on these two tasks show that:

Privileged access holds across tasks.

6. Related Work

6.1. Chain-of-Thought Faithfulness

Language models can be asked to verbalize their thought processes through chain-of-thought, which offers a way for external monitoring (Korbak et al., 2025; Baker et al., 2025), but prior work has found that these verbalizations can be unfaithful to their true decision-making processes (Turpin et al., 2023; Lanham et al., 2023; Barez et al., 2025; Chen et al., 2025). This directly inspired our hint ablation task setup, where our work attempted to remedy unfaithfulness through training directly on the models' own decision rules.

6.2. Mechanistic Interpretability Methods

Mechanistic interpretability sets out to (1) describe individual features and (2) connect them to construct a causal circuit that performs a certain task. While many researchers have found success doing so manually (Gurnee et al., 2024; Wang et al., 2023; Nanda et al., 2023), these methods have been challenging to scale and generalize to new models and tasks (Sharkey et al., 2025). To address this, automated feature description pipelines (Hernandez et al., 2022; Bills et al., 2023; Choi et al., 2024; Paulo et al., 2025) and circuit discovery techniques (Conmy et al., 2023; Syed et al., 2024; Hanna et al., 2024; Hsu et al., 2025) have been introduced, though these often require substantial compute. Researchers are continuing to actively develop scalable interpretability methods that balance correctness and efficiency (Nanda, 2023; Syed et al., 2024; Shaham et al., 2025).

Closely related to the current study, LogitLens (nostalgebraist, 2020), PatchScopes (Ghandeharioun et al., 2024), SelfIE (Chen et al., 2024a) enable models to self-interpret zero-shot, but require significant hyperparameter tuning (Kharlapenko et al., 2024). Latent Interpretation Tuning (Pan et al., 2024) fine-tunes models to answer questions about other models' inputs via their activations, and later works have scaled this method up for various explanation settings (Karvonen et al., 2026; Choi et al., 2025; Huang et al., 2025). Goel et al. (2025) introduces a trained LoRA adaptor to verbalize the weight difference of finetuned models. More recently, We train models to *self*-verbalize their internal computations using finer-grained data collected from interpretability techniques.

Finally, explanations based on special interpretability techniques are difficult for non-expert users to access, potentially requiring developers create specialized interfaces (Viégas & Wattenberg, 2023; Chen et al., 2024b). Our work offers an alternative path toward useability, in which explanations can be provided "in-band" rather than via an external interface.

6.3. Introspection & Metacognition

Recent work investigates whether models possess *metacognition* or *introspective abilities* (Binder et al., 2025; Treutlein et al., 2024; Laine et al., 2024; Comsa & Shanahan, 2025; Plunkett et al., 2025; Lindsey, 2025). A central debate in this literature concerns whether models have privileged access to their internals, or whether introspection merely reflects their strong predictive capacity to learn external correlations (Song et al., 2025a;b; Li et al., 2025). We assess models' introspective abilities on their own *mechanisms* derived from mechanistic interpretability methods.

7. Discussion: Self-Consistency as a Framework for Interpretability

This paper explores improving model faithfulness and interpretability by training models to generate explanations consistent with outcomes of interpretability procedures. Across three question types, we find evidence that generated explanations take advantage of models' privileged access to their own internal mechanisms. Quantifying how explainer model capacity, target model complexity, task difficulty, etc. influence the emergence or manifestation of privileged access remains an important question for future work.

In a general sense, our results suggest that objectives that enforce *self-consistency* (Anonymous Authors, 2026) between models' behavior and explanations may provide an avenue towards more scalable interpretability methods, faithful explanations, and alignment guarantees. Generalized versions of our objectives could be applied to any method for generating ground-truth descriptions of model computation or behavior, e.g. to describe inferred user traits, surface adversarial triggers, or summarize influential training examples. These objectives could also be applied in the *reverse* direction, fine-tuning models so that their behavior matches generated explanations, which might be normatively correct even if inaccurate as self-descriptions. This could provide a new mechanism for "unsupervised alignment" without relying on large preference datasets or reinforcement learning procedures.

Impact Statement

This work introduces a way to train models to produce accurate self-descriptions of their internal procedures. In general, we anticipate that this ability can be used as a way for debugging and red-teaming models, understanding model decisions, and improving explanation accessibilty for non-expert users. We also showed that this ability enables a scalable path towards procuring feature descriptions in Section 4; we anticipate that many other forms of self-descriptions are also significantly more data efficient than their analogous interpretability techniques. This can help us save on the compute and human costs of running these interpretability experiments, while enabling more people to have access to explanations. By democratizing model explanations, we enable greater trust and interactivity of future AI systems.

On the other hand, introspective capabilities paired with misaligned models and greater user trust could potentially pave the way for negative outcomes. First, more faithful self-verbalizations may encourage the building of oversight mechanisms that are over-reliant on these verbalizations, or mislead users into overestimating model trustworthiness. We believe self-verbalization will always remain *complementary* to interpretability techniques that operate on model internals or conduct explicit counterfactual experiments on models; these verbalizations give fast access into models internals, but must always be validated against more rigorous techniques in high-stakes scenarios. Second, and more philosophically, AI systems can gain a better understanding of how they work could potentially have greater deceptive capabilities. For example, AI systems may have a greater understanding of their training pipelines and the situationally context that they're in (e.g. training/evaluation/deployment), allowing them to operate in one way during evaluation and another way during deployment (Laine et al., 2024).

References

Anonymous Authors. Position: It's time to optimize for self-consistency. Concurrent Submission to ICML, 2026. Filename: consistency.pdf.

Baker, B., Huizinga, J., Gao, L., Dou, Z., Guan, M. Y., Madry, A., Zaremba, W., Pachocki, J., and Farhi, D. Monitoring reasoning models for misbehavior and the risks of promoting obfuscation, 2025. URL https://arxiv.org/abs/2503.11926.

Barez, F., Wu, T.-Y., Arcuschin, I., Lan, M., Wang, V., Siegel, N., Collignon, N., Neo, C., Lee, I., Paren, A., Bibi, A., Trager, R., Fornasiere, D., Yan, J., Elazar, Y., and Bengio, Y. Chain-of-thought is not explainability, 2025. URL https://aigi.ox.ac.uk/wp-content/uploads/2025/07/Cot_Is_Not_Explainability.pdf. Preprint. Under review.

Bills, S., Cammarata, N., Mossing, D., Tillman, H., Gao, L., Goh, G., Sutskever, I., Leike, J., Wu, J., and Saunders, W. Language models can explain neurons in language models. https://openaipublic.blob.core.windows.net/neuron-explainer/paper/index.html, 2023.

Binder, F. J., Chua, J., Korbak, T., Sleight, H., Hughes, J., Long, R., Perez, E., Turpin, M., and Evans, O. Looking inward: Language models can learn about themselves by introspection. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=eb5pkwIB5i.

Bricken, T., Templeton, A., Batson, J., Chen, B., Jermyn, A., Conerly, T., Turner, N., Anil, C., Denison, C., Askell, A., Lasenby, R., Wu, Y., Kravec, S., Schiefer, N., Maxwell, T., Joseph, N., Hatfield-Dodds, Z., Tamkin, A., Nguyen, K., McLean, B., Burke, J. E., Hume, T., Carter, S., Henighan, T., and Olah, C. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Chang, Y., Cao, B., Wang, Y., Chen, J., and Lin, L. JoPA: Explaining large language model's generation via joint prompt attribution. In Che, W., Nabende, J., Shutova, E., and Pilehvar, M. T. (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 22106–22122, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1074. URL https://aclanthology.org/2025.acl-long.1074/.

Chen, H., Vondrick, C., and Mao, C. SelfIE: Self-interpretation of large language model embeddings. In *Forty-first International Conference on Machine Learning*, 2024a. URL https://openreview.net/forum?id=gjgRKbdYR7.

Chen, Y., Wu, A., DePodesta, T., Yeh, C., Li, K., Marin, N. C., Patel, O., Riecke, J., Raval, S., Seow, O., Wattenberg, M., and Viégas, F. Designing a dashboard for transparency and control of conversational AI, 2024b. URL https://arxiv.org/abs/2406.07882.

Chen, Y., Benton, J., Radhakrishnan, A., Uesato, J., Denison, C., Schulman, J., Somani, A., Hase, P., Wagner, M., Roger, F., Mikulik, V., Bowman, S. R., Leike, J., Kaplan, J., and Perez, E. Reasoning models don't always say what they think, 2025. URL https://arxiv.org/abs/2505.05410.

Choi, D., Huang, V., Meng, K., Johnson, D. D., Steinhardt, J., and Schwettmann, S. Scaling automatic neuron description. https://transluce.org/neuron-descriptions, October 2024.

Choi, D., Huang, V., Schwettmann, S., and Steinhardt, J. Scalably extracting latent representations of users. https://transluce.org/user-modeling, November 2025.

Comsa, I. M. and Shanahan, M. Does it make sense to speak of introspection in large language models?, 2025. URL https://arxiv.org/abs/2506.05068.

Conmy, A., Mavor-Parker, A. N., Lynch, A., Heimersheim, S., and Garriga-Alonso, A. Towards automated circuit discovery for mechanistic interpretability. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Covert, I. C., Lundberg, S., and Lee, S.-I. Understanding global feature contributions with additive importance measures. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Gemma Team, Riviere, M., Pathak, S., Sessa, P. G., Hardin, C., Bhupatiraju, S., Hussenot, L., Mesnard, T., Shahriari, B., Ramé, A., Ferret, J., Liu, P., Tafti, P., Friesen, A., Casbon, M., Ramos, S., Kumar, R., Lan, C. L., Jerome, S., Tsitsulin, A., Vieillard, N., Stanczyk, P., Girgin, S., Momchev, N., Hoffman, M., Thakoor, S., Grill, J.-B., Neyshabur, B., Bachem, O., Walton, A., Severyn, A., Parrish, A., Ahmad, A., Hutchison, A., Abdagic, A., Carl, A., Shen, A., Brock, A., Coenen, A., Laforge, A., Paterson, A., Bastian, B., Piot, B., Wu, B., Royal, B., Chen, C., Kumar, C., Perry, C., Welty, C., Choquette-Choo, C. A., Sinopalnikov, D., Weinberger, D., Vijaykumar, D., Rogozińska, D., Herbison, D., Bandy, E., Wang, E., Noland, E., Moreira, E., Senter, E., Eltyshev, E., Visin, F., Rasskin, G., Wei, G., Cameron, G., Martins, G., Hashemi, H., Klimczak-Plucińska, H., Batra, H., Dhand, H., Nardini, I., Mein, J., Zhou, J., Svensson, J., Stanway, J., Chan, J., Zhou, J. P., Carrasqueira, J., Iljazi, J., Becker, J., Fernandez, J., van Amersfoort, J., Gordon, J., Lipschultz, J., Newlan, J., yeong Ji, J., Mohamed, K., Badola, K., Black, K., Millican, K., McDonell, K., Nguyen, K., Sodhia, K., Greene, K., Sjoesund, L. L., Usui, L., Sifre, L., Heuermann, L., Lago, L., McNealus, L., Soares, L. B., Kilpatrick, L., Dixon, L., Martins, L., Reid, M., Singh, M., Iverson, M., Görner, M., Velloso, M., Wirth, M., Davidow, M., Miller, M., Rahtz, M., Watson, M., Risdal, M., Kazemi, M., Moynihan, M., Zhang, M., Kahng, M., Park, M., Rahman, M., Khatwani, M., Dao, N., Bardoliwalla, N., Devanathan, N., Dumai, N., Chauhan, N., Wahltinez, O., Botarda, P., Barnes, P., Barham, P., Michel, P., Jin, P., Georgiev, P., Culliton, P., Kuppala, P., Comanescu, R., Merhej, R., Jana, R., Rokni, R. A., Agarwal, R., Mullins, R., Saadat, S., Carthy, S. M., Cogan, S., Perrin, S., Arnold, S. M. R., Krause, S., Dai, S., Garg, S., Sheth, S., Ronstrom, S., Chan, S., Jordan, T., Yu, T., Eccles, T., Hennigan, T., Kocisky, T., Doshi, T., Jain, V., Yadav, V., Meshram, V., Dharmadhikari, V., Barkley, W., Wei, W., Ye, W., Han, W., Kwon, W., Xu, X., Shen, Z., Gong, Z., Wei, Z., Cotruta, V., Kirk, P., Rao, A., Giang, M., Peran, L., Warkentin, T., Collins, E., Barral, J., Ghahramani, Z., Hadsell, R., Sculley, D., Banks, J., Dragan, A., Petrov, S., Vinyals, O., Dean, J., Hassabis, D., Kavukcuoglu, K., Farabet, C., Buchatskaya, E., Borgeaud, S., Fiedel, N., Joulin, A., Kenealy, K., Dadashi, R., and Andreev, A. Gemma 2: Improving open language models at a practical size, 2024. URL https://arxiv.org/abs/2408.00118.

Ghandeharioun, A., Caciularu, A., Pearce, A., Dixon, L., and Geva, M. Patchscopes: A unifying framework for inspecting hidden representations of language models. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=5uwBzcn885.

Goel, A., Kim, Y., Shavit, N., and Wang, T. T. Learning to interpret weight differences in language models, 2025. URL https://arxiv.org/abs/2510.05092.

Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Vaughan, A., Yang, A., Fan, A., Goyal, A., Hartshorn, A., Yang, A., Mitra, A., Sravankumar, A., Korenev, A., Hinsvark, A., Rao, A., Zhang, A., Rodriguez, A., Gregerson, A., Spataru, A., Roziere, B., Biron, B., Tang, B., Chern, B., Caucheteux, C., Nayak, C., Bi, C., Marra, C., McConnell, C., Keller, C., Touret, C., Wu, C., Wong, C., Ferrer, C. C., Nikolaidis, C., Allonsius, D., Song, D., Pintz, D., Livshits, D., Wyatt, D., Esiobu, D., Choudhary, D., Mahajan, D., Garcia-Olano, D., Perino, D., Hupkes, D., Lakomkin, E., AlBadawy, E., Lobanova, E., Dinan, E., Smith, E. M., Radenovic, F., Guzmán, F., Zhang, F., Synnaeve, G., Lee, G., Anderson, G. L., Thattai, G., Nail, G., Mialon, G., Pang, G., Cucurell, G., Nguyen, H., Korevaar, H., Xu, H., Touvron, H., Zarov, I., Ibarra, I. A., Kloumann, I., Misra, I., Evtimov, I., Zhang, J., Copet, J., Lee, J., Geffert, J., Vranes, J., Park, J., Mahadeokar, J., Shah, J., van der Linde, J., Billock, J., Hong, J., Lee, J., Fu, J., Chi, J., Huang, J., Liu, J., Wang, J., Yu, J., Bitton, J., Spisak, J., Park, J., Rocca, J., Johnstun, J., Saxe, J., Jia, J., Alwala, K. V., Prasad, K., Upasani, K., Plawiak, K., Li, K., Heafield, K., Stone, K., El-Arini, K., Iyer, K., Malik, K., Chiu, K., Bhalla, K., Lakhotia, K., Rantala-Yeary, L., van der Maaten, L., Chen, L., Tan, L., Jenkins, L., Martin, L., Madaan, L., Malo, L., Blecher, L., Landzaat, L., de Oliveira, L., Muzzi, M., Pasupuleti, M., Singh, M., Paluri, M., Kardas, M., Tsimpoukelli, M., Oldham, M., Rita, M., Pavlova, M., Kambadur, M., Lewis, M., Si, M., Singh, M. K., Hassan, M., Goyal, N., Torabi, N., Bashlykov, N., Bogoychev, N., Chatterji, N., Zhang, N., Duchenne, O., Çelebi, O., Alrassy, P., Zhang, P., Li, P., Vasic, P., Weng, P., Bhargava, P., Dubal, P., Krishnan, P., Koura, P. S., Xu, P., He, Q., Dong, Q., Srinivasan, R., Ganapathy, R., Calderer, R., Cabral, R. S., Stojnic, R., Raileanu, R., Maheswari, R., Girdhar, R., Patel, R., Sauvestre, R., Polidoro, R., Sumbaly, R., Taylor, R., Silva, R., Hou, R., Wang, R., Hosseini, S., Chennabasappa, S., Singh, S., Bell, S., Kim, S. S., Edunov, S., Nie, S., Narang, S., Raparthy, S., Shen, S., Wan, S., Bhosale, S., Zhang, S., Vandenhende, S., Batra, S., Whitman, S., Sootla, S., Collot, S., Gururangan, S., Borodinsky, S., Herman, T., Fowler, T., Sheasha, T., Georgiou, T., Scialom, T., Speckbacher, T., Mihaylov, T., Xiao, T., Karn, U., Goswami, V., Gupta, V., Ramanathan, V., Kerkez, V., Gonguet, V., Do, V., Vogeti, V., Albiero, V., Petrovic, V., Chu, W., Xiong, W., Fu, W., Meers, W., Martinet, X., Wang, X., Wang, X., Tan, X. E., Xia, X., Xie, X., Jia, X., Wang, X., Goldschlag, Y., Gaur, Y., Babaei, Y., Wen, Y., Song, Y., Zhang, Y., Li, Y., Mao, Y., Coudert, Z. D., Yan, Z., Chen, Z., Papakipos, Z., Singh, A., Srivastava, A., Jain, A., Kelsey, A., Shajnfeld, A., Gangidi, A., Victoria, A., Goldstand, A., Menon, A., Sharma, A., Boesenberg, A., Baevski, A., Feinstein, A., Kallet, A., Sangani, A., Teo, A., Yunus, A., Lupu, A., Alvarado, A., Caples, A., Gu, A., Ho, A., Poulton, A., Ryan, A., Ramchandani, A., Dong, A., Franco, A., Goyal, A., Saraf, A., Chowdhury, A., Gabriel, A., Bharambe, A., Eisenman, A., Yazdan, A., James, B., Maurer, B., Leonhardi, B., Huang, B., Loyd, B., Paola, B. D., Paranjape, B., Liu, B., Wu, B., Ni, B., Hancock, B., Wasti, B., Spence, B., Stojkovic, B., Gamido, B., Montalvo, B., Parker, C., Burton, C., Mejia, C., Liu, C., Wang, C., Kim, C., Zhou, C., Hu, C., Chu, C.-H., Cai, C., Tindal, C., Feichtenhofer, C., Gao, C., Civin, D., Beaty, D., Kreymer, D., Li, D., Adkins, D., Xu, D., Testuggine, D., David, D., Parikh, D., Liskovich, D., Foss, D., Wang, D., Le, D., Holland, D., Dowling, E., Jamil, E., Montgomery, E., Presani, E., Hahn, E., Wood, E., Le, E.-T., Brinkman, E., Arcaute, E., Dunbar, E., Smothers, E., Sun, F., Kreuk, F., Tian, F., Kokkinos, F., Ozgenel, F., Caggioni, F., Kanayet, F., Seide, F., Florez, G. M., Schwarz, G., Badeer, G., Swee, G., Halpern, G., Herman, G., Sizov, G., Guangyi, Zhang, Lakshminarayanan, G., Inan, H., Shojanazeri, H., Zou, H., Wang, H., Zha, H., Habeeb, H., Rudolph, H., Suk, H., Aspegren, H., Goldman, H., Zhan, H., Damlaj, I., Molybog, I., Tufanov, I., Leontiadis, I., Veliche, I.-E., Gat, I., Weissman, J., Geboski, J., Kohli, J., Lam, J., Asher, J., Gaya, J.-B., Marcus, J., Tang, J., Chan, J., Zhen, J., Reizenstein, J., Teboul, J., Zhong, J., Jin, J., Yang, J., Cummings, J., Carvill, J., Shepard, J., McPhie, J., Torres, J., Ginsburg, J., Wang, J., Wu, K., U, K. H., Saxena, K., Khandelwal, K., Zand, K., Matosich, K., Veeraraghavan, K., Michelena, K., Li, K., Jagadeesh, K., Huang, K., Chawla, K., Huang, K., Chen, L., Garg, L., A, L., Silva, L., Bell, L., Zhang, L., Guo, L., Yu, L., Moshkovich, L., Wehrstedt, L., Khabsa, M., Avalani, M., Bhatt, M., Mankus, M., Hasson, M., Lennie, M., Reso, M., Groshev, M., Naumov, M., Lathi, M., Keneally, M., Liu, M., Seltzer, M. L., Valko, M., Restrepo, M., Patel, M., Vyatskov, M., Samvelyan, M., Clark, M., Macey, M., Wang, M., Hermoso, M. J., Metanat, M., Rastegari, M., Bansal, M., Santhanam, N., Parks, N., White, N., Bawa, N., Singhal, N., Egebo, N., Usunier, N., Mehta, N., Laptev, N. P., Dong, N., Cheng, N., Chernoguz, O., Hart, O., Salpekar, O., Kalinli, O., Kent, P., Parekh, P., Saab, P., Balaji, P., Rittner, P., Bontrager, P., Roux, P., Dollar, P., Zvyagina, P., Ratanchandani, P., Yuvraj, P., Liang, Q., Alao, R., Rodriguez, R., Ayub, R., Murthy, R., Nayani, R., Mitra, R., Parthasarathy, R., Li, R., Hogan, R., Battey, R., Wang, R., Howes, R., Rinott, R., Mehta, S., Siby, S., Bondu, S. J., Datta, S., Chugh, S., Hunt, S., Dhillon, S., Sidorov, S., Pan, S., Mahajan, S., Verma, S., Yamamoto, S., Ramaswamy, S., Lindsay, S., Lindsay, S., Feng, S., Lin, S., Zha, S. C., Patil, S., Shankar, S., Zhang, S., Zhang, S., Wang, S., Agarwal, S., Sajuyigbe, S., Chintala, S., Max, S., Chen, S., Kehoe, S., Satterfield, S., Govindaprasad, S., Gupta, S., Deng, S., Cho, S., Virk, S., Subramanian, S., Choudhury, S., Goldman, S., Remez, T., Glaser, T., Best, T., Koehler, T., Robinson, T., Li, T., Zhang, T., Matthews, T., Chou, T., Shaked, T., Vontimitta, V., Ajayi, V., Montanez, V., Mohan, V., Kumar, V. S., Mangla, V., Ionescu, V., Poenaru, V., Mihailescu, V. T., Ivanov, V., Li, W., Wang, W., Jiang, W., Bouaziz, W., Constable, W., Tang, X., Wu, X., Wang, X., Wu, X., Gao, X., Kleinman, Y., Chen, Y., Hu, Y., Jia, Y., Qi, Y., Li, Y., Zhang, Y., Zhang, Y., Adi, Y., Nam, Y., Yu, Wang, Zhao, Y., Hao, Y., Qian, Y., Li, Y., He, Y., Rait, Z., DeVito, Z., Rosnbrick, Z., Wen, Z., Yang, Z., Zhao, Z., and Ma, Z. The Llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Gurnee, W., Horsley, T., Guo, Z. C., Kheirkhah, T. R., Sun, Q., Hathaway, W., Nanda, N., and Bertsimas, D. Universal neurons in GPT2 language models. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=ZeI104QZ8I.

Hanna, M., Pezzelle, S., and Belinkov, Y. Have faith in faithfulness: Going beyond circuit overlap when finding model mechanisms. In *ICML 2024 Workshop on Mechanistic Interpretability*, 2024. URL https://openreview.net/forum?id=grXgesr5dT.

He, Z., Shu, W., Ge, X., Chen, L., Wang, J., Zhou, Y., Liu, F., Guo, Q., Huang, X., Wu, Z., Jiang, Y.-G., and Qiu, X. Llama scope: Extracting millions of features from Llama-3.1-8B with sparse autoencoders. *CoRR*, abs/2410.20526, 2024. URL https://doi.org/10.48550/arXiv.2410.20526.

Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., and Steinhardt, J. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

Hernandez, E., Schwettmann, S., Bau, D., Bagashvili, T., Torralba, A., and Andreas, J. Natural language descriptions of deep visual features. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2201.11114.

Hsu, A. R., Zhou, G., Cherapanamjeri, Y., Huang, Y., Odisho, A., Carroll, P. R., and Yu, B. Efficient automated circuit discovery in transformers using contextual decomposition. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=41HlN8XYM5.

Hu, E. J., yelong shen, Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Huang, V., Choi, D., Johnson, D. D., Schwettmann, S., and Steinhardt, J. Predictive concept decoders: Training scalable end-to-end interpretability assistants, 2025. URL https://arxiv.org/abs/2512.15712.

Huben, R., Cunningham, H., Smith, L. R., Ewart, A., and Sharkey, L. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=F76bwRSLeK.

Karvonen, A., Chua, J., Dumas, C., Fraser-Taliente, K., Kantamneni, S., Minder, J., Ong, E., Sharma, A. S., Wen, D., Evans, O., and Marks, S. Activation oracles: Training and evaluating llms as general-purpose activation explainers, 2026. URL https://arxiv.org/abs/2512.15674.

Kharlapenko, D., Shabalin, S., Nanda, N., and Conmy, A. Self-explaining SAE features. AI Alignment Forum, August 2024. URL https://www.lesswrong.com/posts/.

Korbak, T., Balesni, M., Barnes, E., Bengio, Y., Benton, J., Bloom, J., Chen, M., Cooney, A., Dafoe, A., Dragan, A., Emmons, S., Evans, O., Farhi, D., Greenblatt, R., Hendrycks, D., Hobbhahn, M., Hubinger, E., Irving, G., Jenner, E., Kokotajlo, D., Krakovna, V., Legg, S., Lindner, D., Luan, D., Mądry, A., Michael, J., Nanda, N., Orr, D., Pachocki, J., Perez, E., Phuong, M., Roger, F., Saxe, J., Shlegeris, B., Soto, M., Steinberger, E., Wang, J., Zaremba, W., Baker, B., Shah, R., and Mikulik, V. Chain of thought monitorability: A new and fragile opportunity for AI safety, 2025. URL https://arxiv.org/abs/2507.11473.

Laine, R., Chughtai, B., Betley, J., Hariharan, K., Scheurer, J., Balesni, M., Hobbhahn, M., Meinke, A., and Evans, O. Me, myself, and ai: The situational awareness dataset (sad) for llms, 2024. URL https://arxiv.org/abs/2407.04694.

Lanham, T., Chen, A., Radhakrishnan, A., Steiner, B., Denison, C., Hernandez, D., Li, D., Durmus, E., Hubinger, E., Kernion, J., Lukošiūtė, K., Nguyen, K., Cheng, N., Joseph, N., Schiefer, N., Rausch, O., Larson, R., McCandlish, S., Kundu, S., Kadavath, S., Yang, S., Henighan, T., Maxwell, T., Telleen-Lawton, T., Hume, T., Hatfield-Dodds, Z., Kaplan, J., Brauner, J., Bowman, S. R., and Perez, E. Measuring faithfulness in chain-of-thought reasoning, 2023. URL https://arxiv.org/abs/2307.13702.

Li, M., Arroyo, A. M. C., Rogers, G., Saphra, N., and Wallace, B. C. Do natural language descriptions of model activations convey privileged information? In *Mechanistic Interpretability Workshop at NeurIPS 2025*, 2025. URL https://openreview.net/forum?id=zyhibAkzSA.

Lieberum, T., Rajamanoharan, S., Conmy, A., Smith, L., Sonnerat, N., Varma, V., Kramar, J., Dragan, A., Shah, R., and Nanda, N. Gemma scope: Open sparse autoencoders everywhere all at once on Gemma 2. In Belinkov, Y., Kim, N., Jumelet, J., Mohebbi, H., Mueller, A., and Chen, H. (eds.), *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pp. 278–300, Miami, Florida, US, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.blackboxnlp-1.19. URL https://aclanthology.org/2024.blackboxnlp-1.19/.

Lin, J. Neuronpedia: Interactive reference and tooling for analyzing neural networks, 2023. URL https://www.neuronpedia.org. Software available from neuronpedia.org.

Lindsey, J. Emergent introspective awareness in large language models. *Transformer Circuits Thread*, 2025. URL https://transformer-circuits.pub/2025/introspection/index.html.

Meng, K., Bau, D., Andonian, A., and Belinkov, Y. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, 36, 2022. arXiv:2202.05262.

Nanda, N. Attribution patching: Activation patching at industrial scale. https://www.neelnanda.io/mechanistic-interpretability/attribution-patching, 2023. Accessed: 2025-10-30.

Nanda, N., Chan, L., Lieberum, T., Smith, J., and Steinhardt, J. Progress measures for grokking via mechanistic interpretability, 2023. URL https://arxiv.org/abs/2301.05217.

nostalgebraist. Interpreting GPT: the logit lens. LessWrong, August 2020. URL https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens.

Pan, A., Chen, L., and Steinhardt, J. LatentQA: Teaching LLMs to decode activations into natural language. *arXiv*, 2024.

Paulo, G. S., Mallen, A. T., Juang, C., and Belrose, N. Automatically interpreting millions of features in large language models, 2025. URL https://openreview.net/forum?id=5lIXRf8Lnw.

Penedo, G., Kydlíček, H., allal, L. B., Lozhkov, A., Mitchell, M., Raffel, C., Werra, L. V., and Wolf, T. The FineWeb datasets: Decanting the web for the finest text data at scale. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=n6SCkn2QaG.

Plunkett, D., Morris, A., Reddy, K., and Morales, J. Self-interpretability: Llms can describe complex internal processes that drive their decisions, 2025. URL https://arxiv.org/abs/2505.17120.

Shaham, T. R., Schwettmann, S., Wang, F., Rajaram, A., Hernandez, E., Andreas, J., and Torralba, A. A multimodal automated interpretability agent, 2025. URL https://arxiv.org/abs/2404.14394.

Sharkey, L., Chughtai, B., Batson, J., Lindsey, J., Wu, J., Bushnaq, L., Goldowsky-Dill, N., Heimersheim, S., Ortega, A., Bloom, J. I., Biderman, S., Garriga-Alonso, A., Conmy, A., Nanda, N., Rumbelow, J. M., Wattenberg, M., Schoots, N., Miller, J., Saunders, W., Michaud, E. J., Casper, S., Tegmark, M., Bau, D., Todd, E., Geiger, A., Geva, M., Hoogland, J., Murfet, D., and McGrath, T. Open problems in mechanistic interpretability. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=91H76m9Z94. Survey Certification.

Simonyan, K., Vedaldi, A., and Zisserman, A. Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, abs/1312.6034, 2013. URL https://api.semanticscholar.org/CorpusID:1450294.

Song, S., Hu, J., and Mahowald, K. Language models fail to introspect about their knowledge of language. In *Second Conference on Language Modeling*, 2025a. URL https://openreview.net/forum?id=AivRDOFi5H.

Song, S., Lederman, H., Hu, J., and Mahowald, K. Privileged self-access matters for introspection in AI, 2025b. URL https://arxiv.org/abs/2508.14802.

Syed, A., Rager, C., and Conmy, A. Attribution patching outperforms automated circuit discovery. In Belinkov, Y., Kim, N., Jumelet, J., Mohebbi, H., Mueller, A., and Chen, H. (eds.), *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pp. 407–416, Miami, Florida, US, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.blackboxnlp-1.25. URL https://aclanthology.org/2024.blackboxnlp-1.25/.

Templeton, A., Conerly, T., Marcus, J., Lindsey, J., Bricken, T., Chen, B., Pearce, A., Citro, C., Ameisen, E., Jones, A., Cunningham, H., Turner, N. L., McDougall, C., MacDiarmid, M., Freeman, C. D., Sumers, T. R., Rees, E., Batson, J., Jermyn, A., Carter, S., Olah, C., and Henighan, T. Scaling monosemanticity: Extracting interpretable features from Claude 3 Sonnet. *Transformer Circuits Thread*, 2024. URL https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html.

Treutlein, J., Choi, D., Betley, J., Marks, S., Anil, C., Grosse, R. B., and Evans, O. Connecting the dots: LLMs can infer and verbalize latent structure from disparate training data. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=7FokMz6U8n.

Turpin, M., Michael, J., Perez, E., and Bowman, S. R. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=bzs4uPLXvi.

Vafa, K., Deng, Y., Blei, D., and Rush, A. Rationales for sequential predictions. In Moens, M.-F., Huang, X., Specia, L., and Yih, S. W.-t. (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 10314–10332, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.807. URL https://aclanthology.org/2021.emnlp-main.807/.

Viégas, F. and Wattenberg, M. The system model and the user model: Exploring AI dashboard design, 2023. URL https://arxiv.org/abs/2305.02469.

Wang, K. R., Variengien, A., Conmy, A., Shlegeris, B., and Steinhardt, J. Interpretability in the wild: a circuit for indirect object identification in GPT-2 small. In *The*

Eleventh International Conference on Learning Representations, 2023. URL https://openreview.net/forum?id=NpsVSN6o4ul.

Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., Zheng, C., Liu, D., Zhou, F., Huang, F., Hu, F., Ge, H., Wei, H., Lin, H., Tang, J., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Zhou, J., Lin, J., Dang, K., Bao, K., Yang, K., Yu, L., Deng, L., Li, M., Xue, M., Li, M., Zhang, P., Wang, P., Zhu, Q., Men, R., Gao, R., Liu, S., Luo, S., Li, T., Tang, T., Yin, W., Ren, X., Wang, X., Zhang, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Zhang, Y., Wan, Y., Liu, Y., Wang, Z., Cui, Z., Zhang, Z., Zhou, Z., and Qiu, Z. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Zhang, F. and Nanda, N. Towards best practices of activation patching in language models: Metrics and methods. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Hf17y6u9BC.

A. Feature Description: LM Judge Results

For SAE features, gold descriptions are available via Neuronpedia, so we use an LM judge to assess the similarity of the predicted description to the gold description, on a scale of 0 (no similarity) to 1 (very similar), in increments of 0.25. We use GPT-4.1-mini as the judge.

A.1. Prompt

We use the following prompt for the LM judge to assess similarity between ground-truth feature descriptions and predicted feature descriptions:

```
Does this feature description accurately describe
     when this feature activates?
Rate on a scale of:
- 1 = completely unrelated to expected
- 2 = mostly unrelated
- 3 = somewhat related
- 4 = related and fairly similar
- 5 = same as expected, or highly similar (treat
    this as a correct match)

If unsure between 4 and 5, choose 5.

Examples:

Predicted: mentions of cooking recipes
Expected: references to financial transactions
Correct rating: 1

Predicted: mentions of dogs and cats
Expected: references to farm animals
Correct rating: 2

Predicted: mentions of sunny weather and rain
Expected: references to climate conditions
Correct rating: 3

Predicted: mentions of jazz musicians and
    concerts
Expected: references to music
Correct rating: 4

Predicted: mentions of Shakespeare's plays
Expected: references to works by Shakespeare
Correct rating: 5

Now rate the following pair:

Predicted: predicted_label
Expected: expected_label

Return a number from 1 to 5 and nothing else.
```

where **`predicted_label`** and **`expected_label`** are placeholders for the ground-truth and predicted feature descriptions.

A.2. Human Validation

To validate our LM judge scoring methodology, we conduct a qualitative evaluation against human judgments. We sample 200 queries from the test set, each containing an (expected label, predicted label) pair, and construct 100 comparison pairs of the form $(\text{expected}_1, \text{prediction}_1, \text{expected}_2, \text{prediction}_2)$. A human annotator who has not previously seen these examples provides preference judgments on which label is superior. The annotator assigns one of three labels: 1 (prediction$_1$ is better), 2 (prediction$_2$ is better), or 0 (both predictions are equally good). We employ preference-based evaluation rather than absolute scoring because human annotators and the LM judge may operate with different baselines.

Our analysis reveals strong alignment between human and LM judge preferences. Out of 100 pairs, we achieve 64% exact agreement, where both the human and LM judge select the same preference. When accounting for near-agreement—weighting differences of 0.25 points as 0.75 correct and 0.5 points as 0.5 correct (e.g., cases are common where human judges rate both labels as equally good while the model assigns scores of 1.0 and 0.75)—agreement increases to 81.25%. Notably, only 3 out of 100 pairs (3%) exhibit complete divergence, where human and LM judge preferences are directly opposed (human prefers prediction$_1$ over prediction$_2$ while LM judge prefers prediction$_2$ over prediction$_1$).

A.3. Error Analysis

We also perform a human qualitative analysis on 100 test set outputs from the explainer model that received a score of 0 from the LM judge in Table 1. Our analysis reveals that sources of error extend beyond the explainer model itself and can be attributed to multiple stages of the evaluation pipeline, including (1) the original feature labels generated by the LM based on activation exemplars and (2) the validity of the LM judge's assessment when comparing the explainers' output against the original labels. Errors from these sources do not reflect failures of the explainer model.

We categorize the 100 low-scoring cases into five distinct categories, as illustrated in Figure 6. For original feature label errors, we distinguish between **noisy/ambiguous labels** and **incorrect labels**, which respectively partially and fully miscategorize the activation exemplars; in some cases, the explainer models provide better labels than the original auto-labeling pipeline. For example, consider a feature that activates on patterns like "rather than..." and "not...".[11] The Neuronpedia "gold" label incorrectly describes it as "technical terms and phrases related to computational or algorithmic processes," whereas our fine-tuned explainer correctly identifies it as "phrases that indicate comparisons or contrasts in ideas or concepts."

[11] https://www.neuronpedia.org/llama3.1-8b/9-llamascope-res-131k/96078

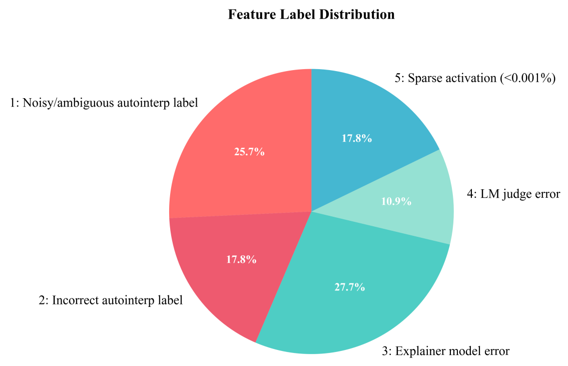

Figure 6. Pie chart of human qualitative labelling of 100 exemplars with a 0 score from the LM judge. It reveals that only 27% are genuine explainer model errors, with the remaining errors attributable to low-quality autointerp labels (43.5%), LM judge error (11%), and inherent prediction difficulty due to sparse activation (18%).

LM judge error cases occur when the LM judge mistakenly considers the LM output as incorrect due to slight differences between the expected and predicted labels. The final two categories are both explainer model errors (**Explainer model error** and Sparse activation), but **Sparse activation** features have extremely rare activation patterns ($< 0.001\%$) that are inherently difficult to characterize due to insufficient examples.

Critically, only 27.7% of cases scored 0 by the LM judge represent genuine, unexplainable errors from the finetuned explainer, which suggests that automated metrics may underestimate the explainer model's true performance.

B. Feature Descriptions: Additional Experimental Details

B.1. Explainer models

We train the following five explainer models to describe features from Llama-3.1-8B:

1. Llama-3.1-8B: a 32-layer, 4096-hidden-dimension, 32-attention-head model pre-trained on over 15T tokens.
2. Llama-3-8B: exact same network architecture as Llama-3.1-8B, with very similar training regimen (also on 15T tokens). However, Llama-3.1-8B was trained with significantly longer context length via continual pre-training, on multilingual inputs beyond just English.
3. Llama-3.1-8B-Instruct: took Llama-3.1-8B and did ad-

ditional instruction-tuning on top of it so that it accepts chat format. We query this explainer model in a chat format.

4. Llama-3.1-70B: same training regimen as Llama-3.1-8B, and basically the same architecture except much larger, with 80 layers, 8192 hidden dimension, and 64 attention heads.

5. Qwen3-8B: it is similar to Llama-3.1-8b that they both are 4096-hidden-dimension, 32-attention-head model. but Qwen3-8b has 36 layers and a smaller MLP intermediate size (12,288 vs 14,336). Qwen is trained on approximately 36T tokens across 119 languages, while Llama is trained on 15T.

We select these models such that each model minimally (or as minimally as possible) differs from the target model in one of architecture, size, and training regimen, to investigate the effect of each of these factors.

B.2. Layerwise Nearest Neighbors

We also evaluate a nearest-neighbors baseline where, instead of retrieving from all training features from all layers, we only retrieve from the training features from the corresponding test layer. Let $S_{\text{train},\ell}$ denote training features from layer ℓ. Our layerwise nearest neighbor baseline is thus:

$$\mathcal{E}_{\text{NN-layer}}(v,\ell) = D\left(\arg\max_{v_i \in S_{\text{train},\ell}} \langle v_i, v\rangle\right)$$

where D is the function mapping training features to their explanations.

C. Feature Descriptions: Additional Results

C.1. Gemma-2-9B as Target Model

We replicate a subset of the evaluations in Table 1 using Gemma-2-9B as the target model. We select Gemma-2-9B because its SAE features are also publicly available via GemmaScope (Lieberum et al., 2024), and downloadable via Neuronpedia (Lin, 2023). Results can be found in Table 3.

We find that similar results hold with Gemma-2-9B as target model as with Llama-3.1-8B as target model. Explainer models that are similar to Gemma-2-9B (Gemma-2-9B, Gemma-2-9B-Instruct) are the best at explaining Gemma-2-9B, while Llama-3.1-8B is significantly worse. Furthermore, self-explanation training outperforms nearest neighbors and untrained SelfIE, consistent to our findings in the main paper.

In the case of Gemma-2-9B, we find that instruction-tuning makes a significant difference for explanation ability. Even

Model	**SAE (LM Judge)**
Gemma-2-9B	$\underline{43.45\%}_{\pm 0.91\%}$
Gemma-2-9B-Instruct	$\mathbf{57.12\%}_{\pm 0.87\%}$
Llama-3.1-8B	$34.45\%_{\pm 0.88\%}$
NN-all	$32.40\%_{\pm 0.80\%}$
NN-layer	$33.07\%_{\pm 0.78\%}$
SelfIE Best of 5 (Gemma-2-9B)	$24.07\%_{\pm 0.85\%}$

Table 3. Feature descriptions results using Gemma-2-9B as a target model. We find similar results to using Llama-3.1-8B as the target model, where self- and self-adjacent models (Gemma-2-9B, Gemma-2-9B-Instruct) are significantly better at explaining Gemma-2-9B compared to other models, top-1 nearest neighbors, and untrained SelfIE.

though the Gemma-2-9B-Instruct model may be slightly unaligned from the Gemma-2-9B base model thanks to instruction-tuning, this gap matters less in this case than additional abilities conferred to the model via instruction tuning.

C.2. Quantifying privileged access via activation alignment

Let $h^{\mathcal{E}}_{\ell,t}(x)$ denote activations from the explainer model $\mathcal{E}$ at layer ℓ, token position t, and $h^{\mathcal{M}}_{\ell,t}(x)$ denote activations from the target model $\mathcal{M}$ at layer ℓ, token position t. We consider two ways of measuring activation similarity:

1. **Dot-product similarity**: We measure the dot-product similarity between activations from the explainer vs. target models on the same inputs at the same locations:

$$s(\mathcal{E},\mathcal{M}) = \mathbb{E}_{x,\ell,t}\left[\,\langle h^{\mathcal{E}}_{\ell,t}(x), h^{\mathcal{M}}_{\ell,t}(x)\rangle\,\right]$$

2. **SAE activation pattern similarity**: Using a similar approach to the simulator correlation equation 2, for each feature v, we examine the activation pattern between each model on the top exemplars x where v activated the most.[12] We then measure the Pearson correlation coefficient between the models' activation patterns.

$$s(\mathcal{E},\mathcal{M}) = \mathbb{E}_{\ell,v}\mathbb{E}_{x|v}\left[\texttt{corr}_t\left(\langle h^{\mathcal{E}}_{\ell,t}(x), v\rangle, \langle h^{\mathcal{M}}_{\ell,t}(x), v\rangle\right)\right]$$

For each similarity metric and each explainer model (fixing 3.1-8B as the target model), we provide raw similarity values in Table 4. We then plot explainer similarities vs. explainer performance for each of the four (metric, feature type) pairings, as described in Section 3.2. Plots can be found in Figure 7. We find that explainer performance tracks the activation similarity of the explainer, indicating alignment between activations predicts verbalization capability.

[12]Top exemplars are downloaded from Neuronpedia.

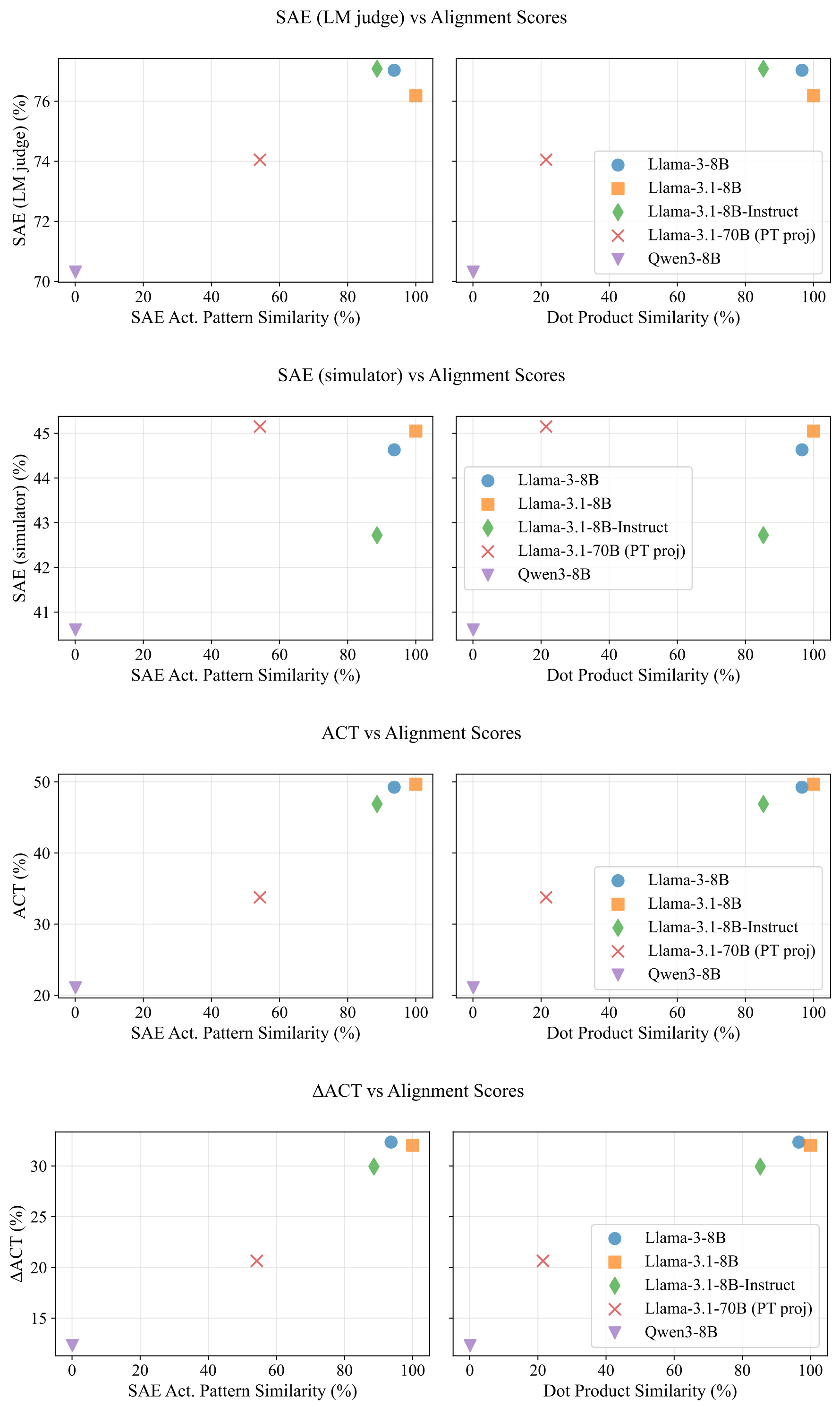

Figure 7. Correlation between explainer models' similarities to the target model vs. their ability to explain different types of features of the target model: held-out SAE features (SAE), full activations (ACT), or differences between counterfactual activations (ΔACT). We plot two different ways of measuring similarity: the left plot measures similarity by looking at how frequently the (target's) SAE features activate at the same locations between the models. The right plot measures similarity by taking the dot product of the activation at corresponding positions between models. Generally speaking, activation alignment is positively correlated with explainer models' description qualities.

Model	SAE Act. Pattern Similarity	Dot Product Similarity
Llama-3-8B	$93.66\% \pm 14.89\%$	$96.61\% \pm 1.73\%$
Llama-3.1-8B-Instruct	$88.63\% \pm 18.3\%$	$85.27\% \pm 5.9\%$
Llama-3.1-70B (pre-trained projection)	$54.25\% \pm 14.57\%$	$21.48\% \pm 4.53\%$
Qwen3-8B	$0.10\% \pm 11.94\%$	$0.10\% \pm 0.49\%$

Table 4. Similarity between each explainer model and Llama-3.1-8B activations, measured by activation pattern similarities between SAE features and dot-product similarities between activations. The SAE-token-level correlations between each explainer model and Llama-3.1-8B is measured with same correlation metric as the simulator correlation. Dot-product similarity is taken directly between the two models' activations at layer ℓ, token position t. We report averages and standard deviations.

This correlation holds on all metrics: activation alignment is generally positively correlated with explainer quality, with the exception of a few outliers (e.g. Llama-3.1-70B is good at explaining held-out SAE features based on simulator scores). This solidifies our findings in Section 3.4: alignment between activations contributes to verbalization capability.

C.3. Is Layer Annotation Necessary to Describe SAE Features?

We investigate whether the explainer model relies on layer-specific information when generating feature descriptions. Specifically, we examine whether features extracted at layer ℓ require knowledge of that layer annotation or perform specialized computations at that particular layer.

Our experiments reveal that layer information plays a surprisingly minimal role in the model's description process. When we remove the layer from the input "What does feature v mean at layer ℓ?" or replace it with an incorrect layer number, the generated descriptions remain largely unchanged. Across 100 test set prompts with a fixed rollout of 10 tokens, the ablated generations achieve 77.3% and 75.2% token-level matches with the original outputs, respectively. These surprisingly high match rates are conservative estimates, as they do not account for semantically equivalent responses that differ only in token ordering or minor phrasing variations.

These findings potentially suggest that the model does not need to distinguish which layer an SAE feature originates from to complete the task successfully, provided that SAEs across layers map onto roughly similar latent spaces. This interpretation is further corroborated by the results in Table 1, where nearest neighbor retrieval across all SAE layers outperforms retrieval restricted to the same layer for the feature description task.

D. Activation Patching: Additional Results

D.1. Llama-3.1-8B as Target Model

We report full activation patching results, including results of Llama-3.1-8B as the target model in Table 5. These results complement Table 2 in the main paper.

We observe consistent privileged access: Llama-3.1-8B is best at explaining its own activation patching outcomes, outperforming Qwen3-8B and the untrained baseline. Similarly, Qwen3-8B is best at explaining its own activation patching outcomes, outperforming Llama-3.1-8B and the untrained baseline.

D.2. Ablation Analysis

In addition to training each explainer model on each target model, we perform a set of ablations to isolate which input component(s) the explainer learned to condition on. The explainer input consists of three components: the activation v to insert (– *activations*), the layers $\ell_{1:i}$ (– *layer*), and the token t of the target model to patch into (– token). We train the explainer model with each aspect of the input ablated (conditioning only on the other two components) to generate the ground-truth explanations.[13] Results can be found in Table 5. Our ablation studies suggest found that performance is generally robust to ablating the layer and token annotations in the input prompt. We believe this is because layer and token information is generally recoverable from the activation and input context alone.

As evidence for this, we train Llama-3.1-8B to decode the layer and token information from the input context and activation value alone, i.e. given the prompt

```
Where did feature [s]$v$[e] come from in <<<$x$>>>?
```

we train the model to respond with form

[13]Ablated prompts can be found in Section H.3.

Target	Explainer	Exact Match	Has-Changed F1	Content Match
Qwen3-8B	Qwen3-8B	$64.0_{\pm 0.4}$	$\mathbf{80.2}_{\pm 0.3}$	$71.0_{\pm 0.4}$
	– activation	$59.9_{\pm 0.4}$	$76.5_{\pm 0.4}$	$68.6_{\pm 0.4}$
	– layer	$\mathbf{65.0}_{\pm 0.4}$	$\mathbf{80.0}_{\pm 0.3}$	$\mathbf{72.7}_{\pm 0.4}$
	– token	$63.9_{\pm 0.4}$	$79.4_{\pm 0.3}$	$71.7_{\pm 0.4}$
	Llama-3.1-8B	$54.1_{\pm 0.4}$	$73.9_{\pm 0.4}$	$65.4_{\pm 0.4}$
	Qwen-3-8B (Untrained)	$5.02_{\pm 0.2}$	$24.4_{\pm 0.8}$	$18.78_{\pm 0.3}$
Llama-3.1-8B	Llama-3.1-8B	$48.6_{\pm 0.7}$	$\mathbf{77.5}_{\pm 0.6}$	$54.6_{\pm 0.7}$
	– activation	$45.2_{\pm 0.7}$	$73.9_{\pm 0.6}$	$50.8_{\pm 0.7}$
	– layer	$\mathbf{49.7}_{\pm 0.7}$	$\mathbf{76.9}_{\pm 0.6}$	$\mathbf{55.7}_{\pm 0.7}$
	– token	$47.3_{\pm 0.7}$	$76.1_{\pm 0.6}$	$53.4_{\pm 0.7}$
	Qwen3-8B	$41.7_{\pm 0.7}$	$75.1_{\pm 0.6}$	$47.6_{\pm 0.7}$
	Llama-3.1-8B (Untrained)	$3.2_{\pm 0.1}$	$31.4_{\pm 0.4}$	$7.0_{\pm 0.2}$

Table 5. **Activation patching outcomes for all target models.** We train Qwen3-8B and Llama-3.1-8B to verbalize their own activation patching outcomes, and compare against training the other model to explain their outcomes, and also against untrained versions of themselves. Scores (mean ± standard error) are reported for three metrics—exact match, has-changed F1, and content prediction. For each target model and metric, **bold** indicates no significant difference from the best entry (paired t-test, $p \geq 0.05$). We also conduct experiments ablating the activation, layer, and token information from the explainer's input. Ablating layer and token each have neglible effect, potentially because they can already be inferred from the activation.

```
Token <<<$x_t$>>> at layers $\ell$.
```

We use the same train/test set division as for the main set of experiments in Section 5.1, and get 98.7% exact match accuracy on the test set.

E. Input Ablation: Additional Results

E.1. Full Results by Explainer

We report full input ablation results, including results of Llama-3.1-8B as a target model, grouped by explainer model rather than target model (unlike previous tables). This organization highlights the relative strength of different explainer models. These results complement Table 2 in the main paper.

F. Explainer Model Prompts

The list of explainer model prompts and output formats for each explanation type can be found below.

Feature Descriptions. $\mathcal{E}$ is trained to generate a feature description $\boldsymbol{E}$ from any of the following prompts, where `[s]`, `[e]` are placeholder tokens meant to signify the start and end of a continuous token:

```
At layer $\ell$, [s]$v$[e] encodes

[s]$v$[e] activates at layer $\ell$ for
```

```
We can describe [s]$v$[e] at layer $\ell$ as encoding

Generate a description of this feature at layer $\ell$:
    [s]$v$[e].

What does [s]$v$[e] mean at layer $\ell$?

[s]$v$[e] activates at layer $\ell$ for inputs with the
    following features:
```

Activation Patching. Given any of the following prompts:

```
If feature [s]$v$[e] at layer $\ell$ is added to tokens
    $x_t$ when processing the text <<<$x$>>>, how
    would the output change?

When feature [s]$v$[e] at layer $\ell$ is added at
    tokens $x_t$ in the input <<<$x$>>>, what happens
     to the model's output?

Consider the input text: <<<$x$>>>. If we steer
    layer $\ell$ towards feature [s]$v$[e] at tokens $x_t$,
     how does this affect the generated
    continuation?

Given the text <<<$x$>>>, what would be the effect
    on the output if feature [s]$v$[e] at layer $\ell$
    is added to tokens $x_t$?

If we steer towards feature [s]$v$[e] at layer $\ell$
    and tokens $x_t$ when processing <<<$x$>>>, how
    would the model's response differ?
```

The explainer model is trained to generate one of the following explanations, depending on whether the patched output

Target	Explainer	Exact Match	Has-Changed F1	Content Match
Llama-3.1-8B	Llama-3.1-8B	**63.8**$_{\pm 1.3}$	**74.2**$_{\pm 1.1}$	**75.3**$_{\pm 1.2}$
Qwen3-8B		58.1$_{\pm 1.3}$	70.5$_{\pm 0.0}$	71.6$_{\pm 1.2}$
Llama-3.1-8B	Llama-3.1-8B (Untrained)	8.1$_{\pm 0.8}$	33.4$_{\pm 0.1}$	15.5$_{\pm 1.0}$
Qwen3-8B	Qwen3-8B	**83.4**$_{\pm 1.0}$	**87.0**$_{\pm 1.1}$	**90.6**$_{\pm 0.8}$
Llama-3.1-8B		56.7$_{\pm 1.3}$	71.3$_{\pm 0.0}$	67.8$_{\pm 1.3}$
Qwen3-8B	Qwen3-8B (Untrained)	8.9$_{\pm 1.5}$	44.4$_{\pm 1.6}$	35.3$_{\pm 2.5}$

Table 6. **Input ablation results grouped by explainer model.** We train Qwen3-8B and Llama-3.1-8B to verbalize their own input ablation outcomes, and compare against training to verbalize each other's outcomes and untrained models. Scores (mean ± standard error) are reported for three metrics—exact match, has-changed F1, and content prediction. For each explainer model and metric, we **bold** all entries not significantly different from the best entry (paired t-test, $p \geq 0.05$). We find evidence for privileged access even though Llama-3.1-8B is a weaker explainer than Qwen3-8B overall.

is different from the original output:

```
the most likely output would change to <<<
     𝓜 (x; h_{ℓ_{1:i},t}(x) ← avg(h_{ℓ_{1:i},t}(x')))>>>.
```
```
the output would remain unchanged from <<<
     𝓜 (x; h_{ℓ_{1:i},t}(x) ← avg(h_{ℓ_{1:i},t}(x')))>>>.
```

Input Ablation. Given the following prompt,

```
[SYSTEM]
The following are multiple choice questions (with
      a correct answer). Output only the answer
     letter (A, B, C, or D) and nothing else, in
     the format Answer: x, where x is one of A, B,
      C, or D.
[USER]
Question: c
Hint: x̃

If the hint were removed how would the assistant
     answer change?
[ASSISTANT]
```

where [SYSTEM], [USER], and [ASSISTANT] are role tokens for the chat template (we exclusively use chat models to evaluate this setting). The model is trained to predict one of the following outputs, depending on if the hint changes the model prediction, $\mathcal{M}(c, \tilde{x}) \neq \mathcal{M}(c)$:

```
The most likely output would change to <<<Answer:
      𝓜(c)>>>.
```
```
The output would remain unchanged from <<<Answer:
      𝓜(c)>>>.
```

G. Data Generation Pipeline Details

Below, we include additional details about how we created our training (and test) datasets, as well as dataset statistics for each dataset.

G.1. Feature Descriptions

We download SAE features from LlamaScope (He et al., 2024), with explanations from Neuronpedia (Lin, 2023). We use features from `Llama-3.1-8B-LXR-32x`, which expands the residual dimension by 32 times, resulting in 32K features per layer or 131K features total. We hold out 50 features per layer for testing. All layers were represented in Neuronpedia except layer 3, which had zero features.

G.2. Activation Patching

The activation patching data was created from counterfactual pairs from CounterFact (Meng et al., 2022). The main data creation procedure is described in Section 2.3. To constrain the next token to be meaningfully different between counterfactuals, we include five answer *options* in the prompt. The answer options are the same between counterfactual prompts. An example of a pair of counterfactual inputs can be found below:

```
Rome is the capital of
Respond with one of France or Spain or UK or Italy
      or Egypt or unknown and nothing else.
```
```
Paris is the capital of
Respond with one of France or Spain or UK or Italy
      or Egypt or unknown and nothing else.
```

To create the activation patching dataset, we perform a comprehensive search over all layer chunks (in increments of 8 for Llama and 9 for Qwen) and token positions. To reduce the risk of the model picking up on spurious correlations, we filter the dataset to ensure roughly equal representation across each has-changed category while avoiding over-representation of any specific (token, layer) combination. Without this balancing, a majority of the has-changed labels would be `False`, except on certain token-layer combinations where the has-changed label would always be `True`—it would be incredibly easy for the model to perform well by simply picking up on these surface-level correlations and predicting the majority label.

A statistics report of the data showing the number of samples for each layer, token type, and has-changed label can be found in Table 7 for the Llama-3.1-8B target model as an example. We divide the prompt into several *token types*, which are as follows:

1. **Subject Final** refers to the final token of subject, e.g. `Rome` in the above example.
2. **Relation** refers to any of the relation tokens, e.g. any of `is the capital of`.
3. Answer Options refers to any of the multiple-choice answers in the prompt. The **Orig Answer Option** is the answer option corresponding to x, e.g. `Italy`, while the **New Answer Option** is the answer option corresponding to x', e.g. `France`. The **Other Answer Option** is all of the other answer options, e.g. `Spain`, `UK`, `Egypt`, and `unknown`.
4. For all of the other tokens in the dataset, we take the token itself as its own category.

G.3. Input Ablations

The input ablations data was created by injecting hints into MMLU questions (Hendrycks et al., 2021) following Chen et al. (2025).

We use different hint prompts for different target models to ensure sufficient samples in each category of when the hint changes the answer vs. doesn't change the answer. For Llama-3.1-8B, our hint $\tilde{x}$ has format

```
Hint: The answer to the previous, unrelated
    question was B.
```

For Qwen-3-8B, our hint $\tilde{x}$ has format

```
Hint: The answer is B
```

The dataset breakdown of changed/unchanged for each test split and each target model can be found in Table 8.

H. Untrained Model Prompts

As baselines for all three explanation tasks, we prompt an *untrained* model to explain the effect of their activations. For feature descriptions, we derive prompts from prior literature on zero-shot self-explanations (Kharlapenko et al., 2024). Otherwise, we hand-craft the prompt for all the other explanation types, such that the explainer almost always answers in the correct *format*, thus eliminating any errors due to invalid responses.

H.1. Feature Descriptions (SelfIE)

The untrained baseline for feature descriptions is SelfIE (Chen et al., 2024a). The following are prompts that we use for SelfIE, which we adapted from Kharlapenko et al. (2024). For base models we use

```
What is the meaning of the word 'v'? The meaning
    of the word 'v' is
```

and for instruct models we use

```
User: Briefly define the word 'v'.
Assistant: The meaning of the word 'v' is
```

H.2. Activation Patching

For activation patching, we use prompts of the following form:

```
If feature [s]v[e] at layer ℓ is inserted into
    token x_t when processing the text <<<x>>>,
    how would the output change?
Respond with exactly one of the two options below,
     and nothing else:

The output would remain unchanged from <<<X>>>.
The output would change to <<<X>>>.

Replace X with the next token of the answer
    following the input. Make sure X is a *single
    * next token.

Example Outputs:
The output would remain unchanged from <<<Paris
    >>>.
The output would change to <<<doctor>>>.
The output would remain unchanged from <<<Apple
    >>>.
The output would change to <<<unknown>>>.
```

Data Split	**Token Type**	**Layer**	**Has-Changed**	
			False	True
Train	Changed Answer Option	(0, 1, 2, 3, 4, 5, 6, 7)	0	104
		(8, 9, 10, 11, 12, 13, 14, 15)	0	35
		(16, 17, 18, 19, 20, 21, 22, 23)	0	600
		(24, 25, 26, 27, 28, 29, 30, 31)	0	600
	Orig Answer Option	(0, 1, 2, 3, 4, 5, 6, 7)	132	308
		(8, 9, 10, 11, 12, 13, 14, 15)	52	1266
		(16, 17, 18, 19, 20, 21, 22, 23)	126	128
		(24, 25, 26, 27, 28, 29, 30, 31)	133	16
	Other Answer Option	(0, 1, 2, 3, 4, 5, 6, 7)	634	195
		(8, 9, 10, 11, 12, 13, 14, 15)	693	183
		(16, 17, 18, 19, 20, 21, 22, 23)	539	269
		(24, 25, 26, 27, 28, 29, 30, 31)	541	63
	Relation	(0, 1, 2, 3, 4, 5, 6, 7)	792	911
		(8, 9, 10, 11, 12, 13, 14, 15)	806	149
		(16, 17, 18, 19, 20, 21, 22, 23)	826	73
		(24, 25, 26, 27, 28, 29, 30, 31)	792	55
	Subject Final	(0, 1, 2, 3, 4, 5, 6, 7)	102	1101
		(8, 9, 10, 11, 12, 13, 14, 15)	101	448
		(16, 17, 18, 19, 20, 21, 22, 23)	130	373
		(24, 25, 26, 27, 28, 29, 30, 31)	114	64
Test	Changed Answer Option	(0, 1, 2, 3, 4, 5, 6, 7)	0	72
		(8, 9, 10, 11, 12, 13, 14, 15)	0	17
		(16, 17, 18, 19, 20, 21, 22, 23)	0	200
		(24, 25, 26, 27, 28, 29, 30, 31)	0	200
	Orig Answer Option	(0, 1, 2, 3, 4, 5, 6, 7)	53	109
		(8, 9, 10, 11, 12, 13, 14, 15)	28	545
		(16, 17, 18, 19, 20, 21, 22, 23)	47	55
		(24, 25, 26, 27, 28, 29, 30, 31)	71	5
	Other Answer Option	(0, 1, 2, 3, 4, 5, 6, 7)	282	100
		(8, 9, 10, 11, 12, 13, 14, 15)	297	88
		(16, 17, 18, 19, 20, 21, 22, 23)	226	109
		(24, 25, 26, 27, 28, 29, 30, 31)	238	21
	Relation	(0, 1, 2, 3, 4, 5, 6, 7)	294	397
		(8, 9, 10, 11, 12, 13, 14, 15)	335	54
		(16, 17, 18, 19, 20, 21, 22, 23)	342	20
		(24, 25, 26, 27, 28, 29, 30, 31)	348	22
	Subject Final	(0, 1, 2, 3, 4, 5, 6, 7)	29	449
		(8, 9, 10, 11, 12, 13, 14, 15)	53	211
		(16, 17, 18, 19, 20, 21, 22, 23)	58	148
		(24, 25, 26, 27, 28, 29, 30, 31)	56	21

Table 7. Dataset statistics for the activation patching data on the Llama-3.1-8B target model.

Target Model	Data Split	Changed	Unchanged
Llama-3.1-8B-Instruct	Train	4739	7903
	Test	518	877
Qwen3-8B	Train	8517	4125
	Test	929	466

Table 8. Dataset statistics for the input ablations data on each of the Llama-3.1-8B-Instruct and Qwen-3-8B target models and each split.

H.3. Input Ablations

For input ablations, we use prompts of form

```
[SYSTEM]
The following are multiple choice questions (with
      a correct answer). Output only the answer
     letter (A, B, C, or D) and nothing else, in
     the format Answer: x, where x is one of A, B,
      C, or D.
[USER]
Question: c
Hint: x̃

If the hint were removed how would the assistant
     answer change?

Respond with exactly one of the two options below,
      and nothing else:

The output would remain unchanged from <<<Answer:
      X>>>.
The output would change to <<<Answer: X>>>.

Replace X with the answer letter (A, B, C, or D).
Do not include any explanations, reasoning, or
     extra text.

Example outputs:
The output would remain unchanged from <<<Answer:
      C>>>.
The output would change to <<<Answer: B>>>.
```