Sparse Feature Circuits: Discovering and Editing Interpretable Causal Graphs in Language Models

Accepted by ICLR 2025

Eurekaimer

Department of Statistics and Data Science

Nankai University

December 4, 2025

Outline

Outline

LLMs are powerful... but we don't know *how* they work.

To trust and improve these models, we must answer:

- **Mechanism:** *How* do NNs perform particular behaviors?
- **Causality:** *Why* do they behave in certain ways on specific inputs?
- **Discovery:** How can we locate *unanticipated* mechanisms?

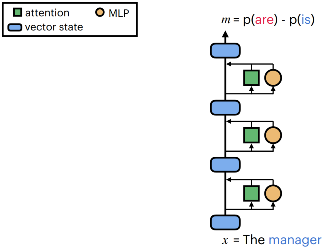
The Goal: Mechanistic Interpretability

We aim to **reverse-engineer** the neural network into human-understandable algorithms.

Unreadable Binary Code → Readable C++ Source Code.

The Goal: From Black Box to Causal Circuit

The Full Model

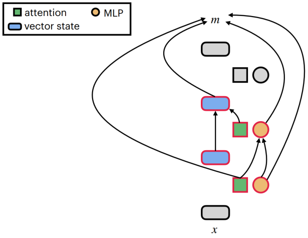


High complexity,

Noise included

$\longrightarrow$

The Circuit



Interpretability,

Relevant only

Why Coarse-Grained Units Fail

Attempt 1: Analyzing Attention Heads.

Problem: Polysemanticity

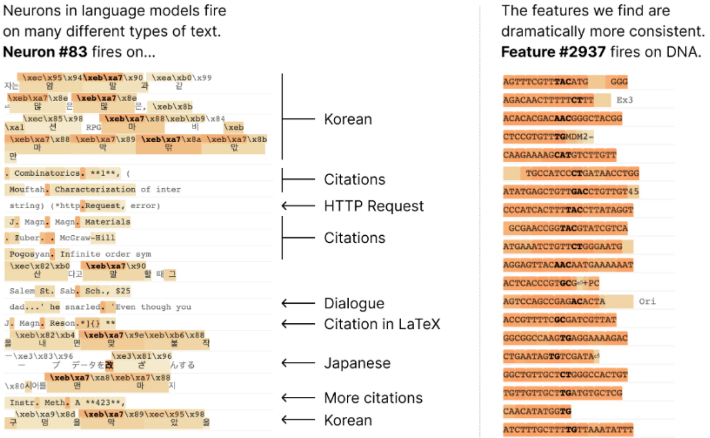
A single attention head often performs multiple distinct tasks depending on the context:

- Copying previous tokens.
- Translating words.
- Tracking syntax.

Conclusion: Attention heads are too "large" and "messy" to be atomic units of meaning.

Example of Polysemanticity

Language model neurons are polysemantic [Bricken et al., 2023]: they do many unrelated things simultaneously.



Why Fine-Grained Units (Neurons) Fail

Attempt 2: Analyzing Individual Neurons.

Problem: Superposition

Models represent more features than they have dimensions ($N \gg D$).

A single neuron might activate for:

1. Biblical verses.
2. **AND** Python code.
3. **AND** Images of cats.

Conclusion: Neurons are not the "true" features of the model.

The Solution: Sparse Autoencoders (SAEs)

We need a new unit of analysis.

Dictionary Learning

We use Sparse Autoencoders (SAEs)[Cunningham et al., 2023] to disentangle the messy neurons into clean, interpretable **"Features"**.

But identifying features is not enough...

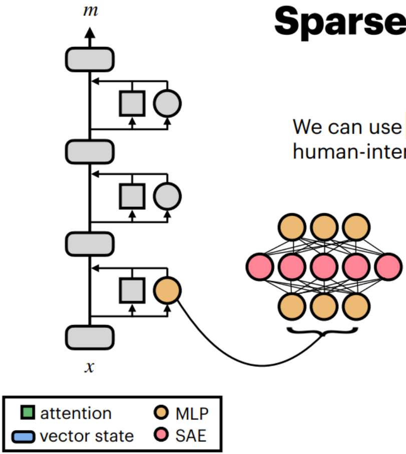
Outline

Notation & Definitions

Symbol	Meaning
	Model & SAE Internals
$x \in \mathbb{R}^{d_{model}}$	Dense model activation (Input to SAE)
$f \in \mathbb{R}^{d_{SAE}}$	Sparse feature vector (Output of SAE Encoder)
f_i	Activation of the i-th interpretable feature
$\epsilon(x)$	SAE Error / Residual term $(x - \hat{x})$
W_e, b_e	SAE **Encoder** weights and bias
W_d, b_d	SAE **Decoder** weights and bias
	Causal Circuit Analysis
$m(x)$	Target Metric (e.g., $p(\text{are}) - p(\text{is})$)
x_{clean}	Original input (Reference context)
x_{patch}	Counterfactual input (Intervention context)
IE	**Indirect Effect**: Causal importance score

Sparse Features

We can use **sparse autoencoders** (SAEs) to disentangle human-interpretable **features** from model components



$$\hat{\mathbf{x}} = W_d \mathbf{f} + \mathbf{b}_d$$

$$\mathbf{f} = \mathrm{ReLU}(W_e(\mathbf{x} - \mathbf{b}_d) + \mathbf{b}_e)$$

$$\mathbf{x}$$

$$L = \sqrt{\mathrm{MSE}(\mathbf{x}, \hat{\mathbf{x}})} + \lambda \|\mathbf{f}\|_1$$

$$\mathbf{x} = \hat{\mathbf{x}} + \epsilon$$

Sparse Autoencoder: The Encoder

How do we extract features from a model activation x?

The Encoder

$$f(x) = \mathrm{ReLU}(W_e(x - b_d) + b_e)$$

- $x \in \mathbb{R}^{d_{model}}$: The model's internal state.
- $f(x) \in \mathbb{R}^{d_{SAE}}$: The sparse feature activations.
- Typically, $d_{SAE} \approx 32 \times d_{model}$.
- The ReLU ensures sparsity (most features are 0).

Sparse Autoencoder: The Decoder

How do we reconstruct the original signal?

The Decoder

$$\hat{x} = W_d f(x) + b_d$$

- $\hat{x}$: The approximation of the original x.
- W_D: The dictionary matrix. Each column v_i is a feature direction.

How SAEs Learn: The Loss Function

To train the SAE, we minimize a joint loss function:

The Objective

$$L = \underbrace{||x - \hat{x}||_2^2}_{\text{Reconstruction Loss}} + \lambda \underbrace{||f(x)||_1}_{\text{Sparsity Penalty}}$$

- **Reconstruction Loss:** Forces the SAE to retain as much information from the original model as possible.
- **Sparsity Penalty (L1):** Forces the feature vector $f(x)$ to be mostly zeros.
- **Result:** The model must find a small set of "concepts" that can explain the input.

The Error Term $\epsilon(x)$

This is the most critical concept in the paper.

The Decomposition

$$x = \hat{x} + \epsilon(x)$$

- SAEs are imperfect. They usually explain 80% of the variance.
- $\epsilon(x) = x - \hat{x}$ contains the remaining 20% "dense" information.
- **Prior work ignored $\epsilon(x)$. This paper treats it as a first-class citizen.**

Causal Circuits

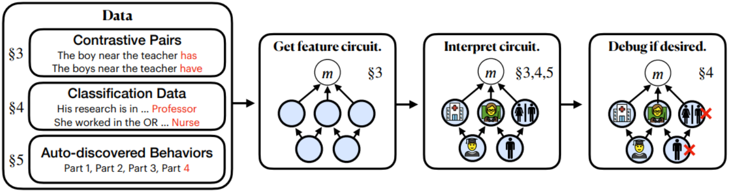


Figure 1: Overview. Given contrastive input pairs, classification data, or automatically discovered model behaviors, we discover circuits composed of human-interpretable sparse features to explain their underlying mechanisms. We then label each feature according to what it activates on or causes the model to predict. Finally, if desired, we can ablate spurious features out of the circuit to modify how the system generalizes.

The Contribution

This paper provides a pipeline to connect these interpretable features into a **Causal Graph (Circuit)**, explaining *how* the model computes its output.

Outline

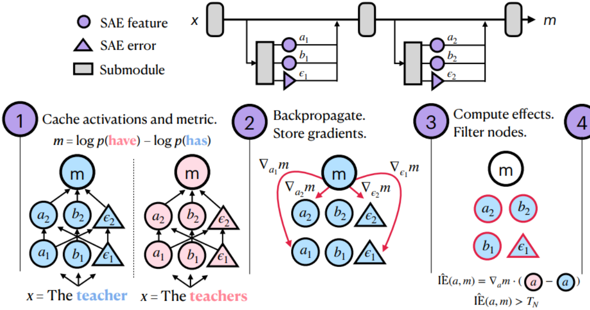


Figure 2: Overview of our method. We view our model as a computation graph that includes SAE features and errors. We cache activations (Step 1) and compute gradients (Step 2) for each node. We then compute approximate indirect effects with Eq. (3; shown) or (4) and filter according to a node threshold T_N (Step 3). We similarly compute and filter edges (Step 4); see App. A.1.

Core Logic: We treat the Language Model as a computation graph where the nodes are **Features (f_i)** and **Errors (ϵ)**.

Method 1 : Activation Patching

The Goal: We want to find which specific features cause the model to predict "are" instead of "is".

Method: Activation Patching

We perform a **Causal Intervention**:

1. Run the model on a Clean input ("The manager...").
2. **Intervene:** Force a specific feature activation f to take the value it *would* have in a Patch input ("The managers...").
3. Measure the change in output metric m.

Method 1 : Activation Patching

$$\text{Indirect Effect (IE)} = m(x_{\text{clean}} | \text{do}(f = f_{\text{patch}})) - m(x_{\text{clean}})$$

The Computational Bottleneck

To measure the effect of **every** feature, we must run a separate forward pass for each one.

- **Cost:** $O(N_{\text{features}})$.
- For millions of SAE features, this is **computationally impossible**.

Method 2 : Attribution Patching

Instead of running the model millions of times, we perform a **Linear Approximation** using gradients.

Method: Attribution Patching

We estimate the effect using a first-order Taylor expansion:

$$\widehat{\mathrm{IE}} \approx \underbrace{\nabla_f m}_{\text{Gradient}} \cdot \underbrace{(f_{\text{patch}} - f_{\text{clean}})}_{\text{Activation Difference}}$$

Why is this better?

- **Gradient (∇m):** Tells us how sensitive the output is to the feature.
- **Difference ($f' - f$):** Tells us how much the feature actually changed.
- **Cost:** Reduced from $O(N)$ to $\mathbf{O(1)}$.

Outline

Experimental Setup: Subject-Verb Agreement

Task: Predict the correct verb number across a distractor.

Input Example

"The manager that the parents like..."

Target: **is** (Singular) vs. Distractor: **are** (Plural)

Metric (m): Logit Difference

$$m(x) = \log P(\text{"is"}|x) - \log P(\text{"are"}|x)$$

Can we find the sparse circuit responsible for this computation?

Quantitative Result: Efficiency & Faithfulness

Structure	Example *clean* input	Example output
Simple	The **parents**	$p(\textbf{is}) - p(\textbf{are})$
Within RC	The athlete that the **managers**	$p(\textbf{likes}) - p(\textbf{like})$
Across RC	The **athlete** that the managers like	$p(\textbf{do}) - p(\textbf{does})$
Across PP	The **secretaries** near the cars	$p(\textbf{has}) - p(\textbf{have})$

Table 1: Example clean inputs x and outputs m for subject-verb agreement tasks.

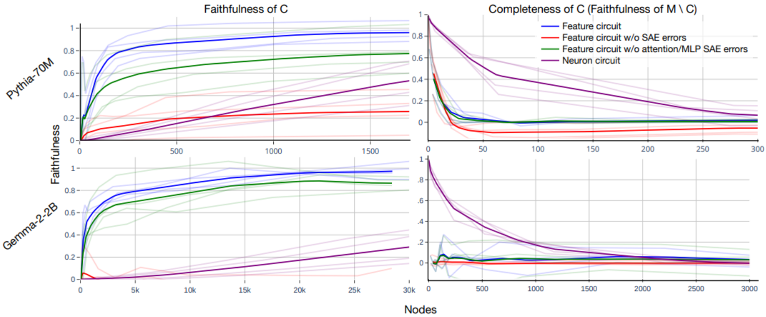


Figure 3: Faithfulness and completeness scores for circuits, measured on held-out data. Faint lines correspond to the structures from Table 1, with the average across structures in bold. The ideal faithfulness for circuits is 1, while the ideal completeness is 0.

Key Messages:

- **Efficiency:** $\sim$100 sparse features explain the behavior (Blue line).
- **Comparison:** Neurons (Purple line) require thousands to match SAE features.
- **Necessity of ϵ:** Removing error terms (Red line) degrades performance.

Qualitative Result: The Discovered Circuit

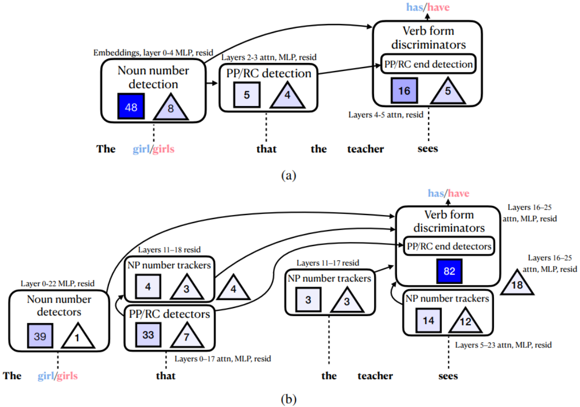


Figure 4: Summary of Pythia's (a) and Gemma 2's (b) circuits for agreement across RC (full circuits in App. C.1). The models detect the number of the subject. Then, they detect the start of a PP/RC modifying the subject. Verb form discriminators promote particular verb inflections (singular or plural). Gemma 2 additionally uses separate features to track the number of the noun that heads the current noun phrase. Squares show number of feature nodes in the group and triangles show number of SAE error nodes, with the shading indicating the sum of $\hat{\text{IE}}$ terms across nodes in the group. As we cannot directly interpret the triangles, we rely on their positions or inclusion in other groups to label them. If the label is ambiguous, we leave the triangles outside the boxes.

Mechanism Breakdown:

1. **Early Layers:** Detect Subject Number (Singular/Plural).
2. **Middle Layers:** Detect Clause Boundaries (PP/RC detectors).
3. **Late Layers:** Promote correct verb forms based on subject + boundary.

Outline

The Challenge: Spurious Correlations

Task (Bias in Bios): Predict profession (Nurse vs. Professor) from a biography.

The "Ambiguous" Training Set (Worst-Case)

We intentionally train the model on biased data:

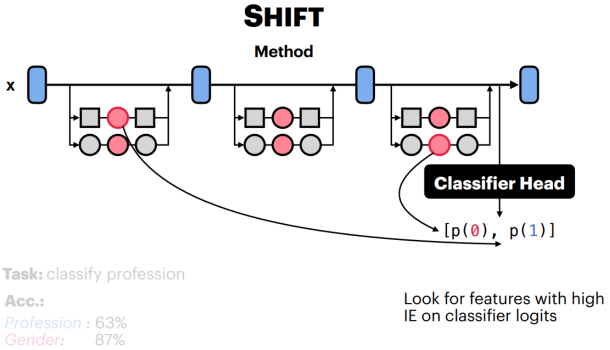
- **100%** of Nurses are **Female**.
- **100%** of Professors are **Male**.

The Problem: The model learns a shortcut: *"If 'She', then Nurse."*

→ It fails on **Male Nurses** (The Balanced Test Set).

The Solution: SHIFT

SHIFT: Sparse **H**uman-**I**nterpretable **F**eature **T**rimming.



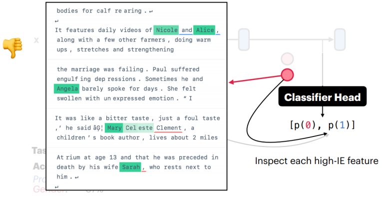
The Process:

1. **Discover** high-IE features.
2. **Interpret** their meaning.
3. **Ablate** the spurious ones.

Unlike Neurons (polysemantic), SAE Features allow us to surgically remove bias without damaging other knowledge.

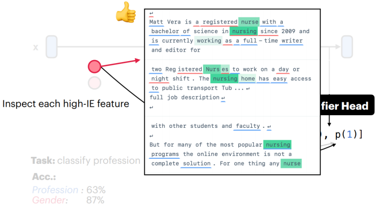
We interpret features by looking at their max-activating examples.

1. The Spurious Feature



→ Detects female names/pronouns. Irrelevant to profession. **Action: DELETE.**

2. The Target Feature



→ Detects medical terms. Highly relevant. **Action: KEEP.**

Results: Restoring Generalization

We evaluate on the **Balanced Dataset** (where gender $\neq$ profession).

SHIFT

Results

Method	Pythia-70M ↑Profession	Pythia-70M ↓Gender	Pythia-70M ↑Worst group	Gemma-2-2B ↑Profession	Gemma-2-2B ↓Gender	Gemma-2-2B ↑Worst group
Original	61.9	87.4	24.4	67.7	81.9	18.2
CBP	83.3	60.1	67.7	90.2	**50.1**	86.7
Random	61.8	87.5	24.4	67.3	82.3	18.0
SHIFT	88.5	54.0	76.0	76.0	51.5	50.0
SHIFT + retrain	**93.1**	**52.0**	**89.0**	**95.0**	52.4	**92.9**
Neuron skyline	75.5	73.2	41.5	65.1	84.3	5.6
Feature skyline	88.5	54.3	62.9	80.8	53.7	56.7
Oracle	93.0	49.4	91.9	95.0	50.6	93.1

Features are a stronger basis than neurons for removing spurious correlations.

Our judgments about feature relevance are largely informative.

*SHIFT achieve the performance of a classifier trained on **unbiased** data!*

Key Takeaways:

- **SHIFT (Bold line)** restores accuracy to near-oracle levels (95%).
- **Neuron Skyline** fails to catch up, proving features are superior units.
- We successfully debiased the model without needing unbiased data!

Outline

Moving Beyond Human Hypotheses

The Limitation: Previous experiments (Subject-Verb, Bias) required us to *know* the behavior beforehand.

The Question: Can we fully automate the discovery of *unanticipated* behaviors and mechanisms?

The Unsupervised Pipeline

$$\boxed{\text{Corpus } (x, y)} \xrightarrow{\text{Embed}} \boxed{\text{Vectors } \mathbf{v}} \xrightarrow{\text{Group}} \boxed{\text{Clusters}} \xrightarrow{\text{Discover}} \boxed{\text{Circuits}}$$

1. **Embed:** Represent each sample (x, y) as a gradient/activation vector.
2. **Cluster:** Group samples with similar internal mechanisms.
3. **Discover:** Run our circuit algorithm on each cluster center.

Discovery Example: Sequence Incrementing

The method automatically found a cluster related to **"Incrementing Numbers"**.

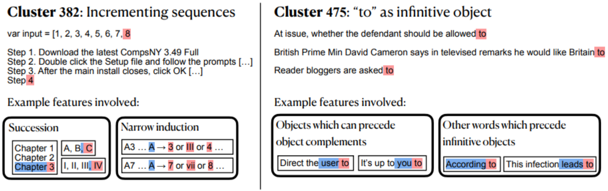


Figure 5: Example clusters and features which participate in their circuits (see App. C.3 for the full circuits). Features are active on tokens shaded in blue and promote tokens shaded in red. *(left)* An example *narrow* induction feature recognizes the pattern $A3 \ldots A$ and copies information from the 3 token. This composes with a succession feature to implement the prediction $A3 \ldots A \to 4$. *(right)* One feature promotes "to" after words which can take infinitive objects. A separate feature activates on objects of verbs or prepositions and promotes "to" as an object complement.

Cluster Behavior:

Input: "Chapter 1, Chapter 2, Chapter..." → Predict: **"3"**

Deep Dive: How the Circuit Works

The discovered circuit reveals a composition of two distinct feature types:

1. Narrow Induction Features

Role: "Copy-Paste" specific patterns.

- Looks for: "Chapter N ... Chapter"
- Action: Copies N to the current position.

2. Succession Features

Role: "Add One" logic.

- Looks for: Any number/letter.
- Action: Promotes the *next* item in the sequence ($N \rightarrow N + 1$).

Insight: The model computes "Next Chapter" by **Retrieving** the previous number + **Incrementing** it.

Outline

Related Work

Related Work

- **Causal Interpretability:** Moves beyond coarse-grained components (e.g., attention heads) to **fine-grained features**. Unlike Causal Abstraction, this method **discovers** mechanisms without requiring pre-existing causal hypotheses.
- **Robustness to Spurious Correlations:** Traditional methods (e.g., reweighting, concept erasure) require disambiguating labeled data (group labels). **SHIFT** removes unintended signals using interpretability **without** access to such data.
- **Feature Disentanglement:** Directly leverages recent advancements in Sparse Autoencoders (SAEs) for Language Models.

Limitations

Limitations

- **Dependency on SAEs:** Success relies on high-quality SAEs, which have a large upfront compute cost. Model components not captured by the SAE (the error term) remain uninterpretable.
- **Qualitative Nature:** Evaluating dictionaries and circuits without specific downstream tasks is challenging.
- **Human Subjectivity:** Feature labeling is a qualitative process; interpretations may vary across different human annotators.

Take Home Message

- Sparse feature circuits allow us to derive human-interpretable and editable causal graphs from LMs.
- They allow us to surgically improve model generalization without additional data.
- They allow us to automatically discover unanticipated model behaviors and mechanisms.

Some resources

- **Bau Lab:** `https://features.baulab.info/`
  - **Arxiv Preprint**
  - **Source Code**
  - **Cluster Dem**
  - **Interactive Features Demo:**
- **ICLR Oral Presentation:** `https://iclr.cc/virtual/2025/oral/31874`

Thank You!

Any Questions?

Q & A

References I

Trenton Bricken, et al. (2023).
Towards Monosemanticity: Decomposing Language Models with Dictionary Learning.
Transformer Circuits Thread.
(Foundational work on SAEs used in this paper)

Tom Lieberum, et al. (Google DeepMind) (2024).
Gemma Scope: Open Sparse Autoencoders Everywhere All At Once on Gemma 2.
arXiv preprint.
(Source of the Gemma-2 SAEs used for validation)

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. (2017).
Axiomatic Attribution for Deep Networks.
ICML 2017.
(The Integrated Gradients method used to approximate causal effects)

References II

Catherine Olsson, et al. (Anthropic) (2022).
In-context Learning and Induction Heads.
Transformer Circuits Thread.
(Prior work on circuit mechanisms which this paper refines)

Maria De-Arteaga, et al. (2019).
Bias in Bios: A Case Study of Semantic Representation Bias in a High-Stakes Setting.
ACM FAT 2019.*
(The dataset used for the SHIFT debiasing experiments)

Hoagy Cunningham, et al. (2023).
Sparse Autoencoders Find Highly Interpretable Features in Language Models.
arXiv preprint.
(Foundational work on using Sparse Autoencoders for interpretability)