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

**Samuel Marks**[*] Northeastern University

**Can Rager** Independent

**Eric J. Michaud** MIT

**Yonatan Belinkov** Technion – IIT

**David Bau** Northeastern University

**Aaron Mueller**[*] Northeastern University

# MI2 Seminar

Dawid Płudowski

March 17th, 2025

# Authors


**Samuel Marks**

**Can Rager**

**Eric J. Michaud**

**Yonatan Belinkov**

**David Bau**

**Aaron Mueller**

# Authors

## Samuel Marks

OBSERWUJ

Postdoctoral researcher, Northeastern University
Zweryfikowany adres z northeastern.edu - Strona główna

large language models  interpretability  AI safety

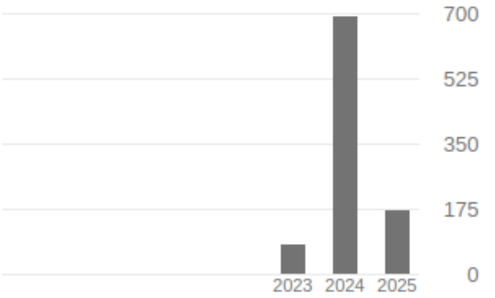
**10+ articles in 2024**

| TYTUŁ | CYTOWANE PRZEZ | ROK |
|---|---|---|
| Open problems and fundamental limitations of reinforcement learning from human feedback<br>S Casper, X Davies, C Shi, TK Gilbert, J Scheurer, J Rando, R Freedman, ...<br>arXiv preprint arXiv:2307.15217 | 499 | 2023 |
| The geometry of truth: Emergent linear structure in large language model representations of true/false datasets<br>S Marks, M Tegmark<br>arXiv preprint arXiv:2310.06824 | 124 | 2023 |
| The wmdp benchmark: Measuring and reducing malicious use with unlearning<br>N Li, A Pan, A Gopal, S Yue, D Berrios, A Gatti, JD Li, AK Dombrowski, ...<br>arXiv preprint arXiv:2403.03218 | 120 | 2024 |
| Sparse feature circuits: Discovering and editing interpretable causal graphs in language | 87 | 2024 |

UTWÓRZ SWÓJ PROFIL

Cytowane przez

| | Wszystkie | Od 2020 |
|---|---|---|
| Cytowania | 950 | 949 |
| h-indeks | 9 | 9 |
| i10-indeks | 8 | 8 |



# Article

Base info

- Northeastern
- 80+ citations
- Preprint on Arxiv,
- ~03.2024

Contribution

- scalable method to discover sparse circuits
- new method, SHIFT
- detecting *feature circuits* in (un)supervised schema

# Motivation

*Circuits identified in prior work consist of polysemantic and difficult-to-interpret units like attention heads or neurons, rendering them unsuitable for many downstream applications. In contrast, sparse feature circuits enable detailed understanding of unanticipated mechanisms. Because they are based on fine-grained units, sparsefeature circuits are useful for downstream tasks: We introduce SHIFT, where we improve the generalization of a classifier by ablating features that a human judges to be task-irrelevant.*

- **So far, we:**
  - know what SAE is
  - know how to detect if SAE features are polysemantic or not
- **Now, we learn:**
  - how to use them in LLMs
  - how to detect feature circuits in models

# SAE in the NN

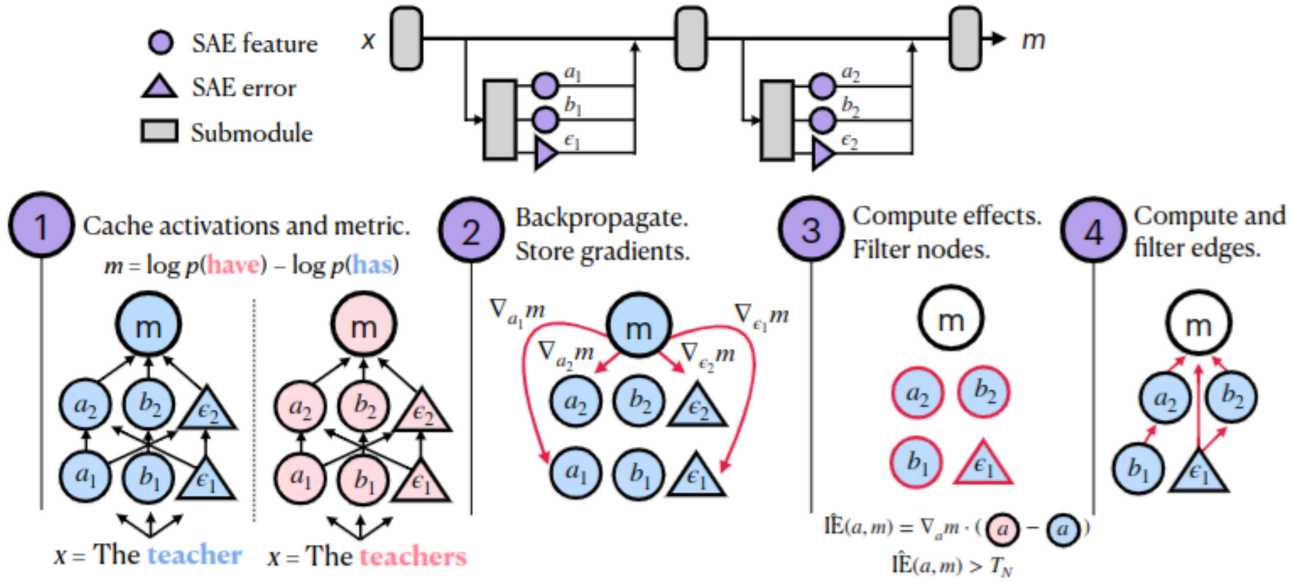



Figure 2: Overview of our method. We view our model as a computation graph that includes SAE features and errors. We cache activations (Step 1) and compute gradients (Step 2) for each node. We then compute approximate indirect effects with Eq. (3; shown) or (4) and filter according to a node threshold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

# How to create a circuit?

- *m* - arbitrary metric (in the article - p(y|x))
- **a** - a node in a computational graph (in the article - a SAE feature)
- *x* - an input
- do(**a** = **a**_patch) - replace activation of **a** as if the input is *x_path*, not *x_clean*

$$\mathrm{IE}(m; \mathbf{a}; x_{\text{clean}}, x_{\text{patch}}) = m\left(x_{\text{clean}} \middle| \mathrm{do}(\mathbf{a} = \mathbf{a}_{\text{patch}})\right) - m(x_{\text{clean}}).$$

# IE for edges

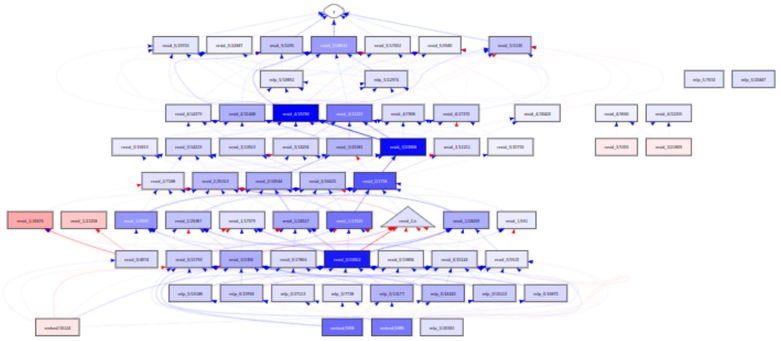
(a) $e$ connects a layer $\ell$ residual stream component and a layer $\ell$ attention or MLP component, or $e$ connects a layer $\ell$ attention or MLP component and a layer $\ell+1$ residual stream component;[4]

$$\begin{aligned}\hat{\mathrm{IE}}(m; e; x_{\text{clean}}, x_{\text{patch}}) &= \nabla_{\mathbf{d}} m\big|_{\mathbf{d}_{\text{clean}}} \nabla_{\mathbf{u}} \mathbf{d}\big|_{\mathbf{u}_{\text{clean}}} \left(\mathbf{u}_{\text{patch}} - \mathbf{u}_{\text{clean}}\right) \\ &\approx m\left(x_{\text{clean}} \middle| \text{do}\left(\mathbf{d} = \mathbf{d}\left(x_{\text{clean}} \middle| \text{do}\left(\mathbf{u} = \mathbf{u}_{\text{patch}}\right)\right)\right)\right)\end{aligned}$$

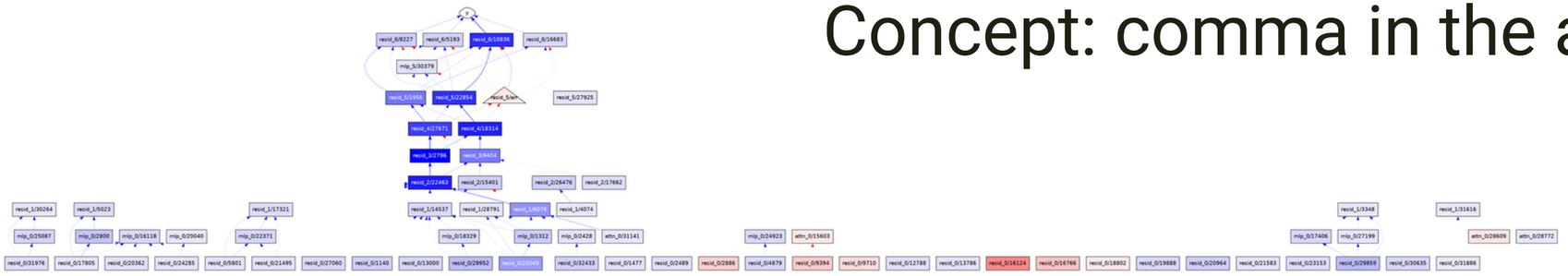
(b) $e$ connects a layer $\ell$ residual stream component and a layer $\ell+1$ residual stream component.

$$\hat{\mathrm{IE}}(m; e; x_{\text{clean}}, x_{\text{patch}}) = \nabla_{\mathbf{d}} m\big|_{\mathbf{d}_{\text{clean}}} \left(\nabla_{\mathbf{u}} \mathbf{d}\big|_{\mathbf{u}_{\text{clean}}} - \sum_{\mathbf{m}} \nabla_{\mathbf{m}} \mathbf{d}\big|_{\mathbf{m}_{\text{clean}}} \nabla_{\mathbf{u}} \mathbf{m}\big|_{\mathbf{u}_{\text{clean}}}\right) \left(\mathbf{u}_{\text{patch}} - \mathbf{u}_{\text{clean}}\right)$$

# Feature circuits

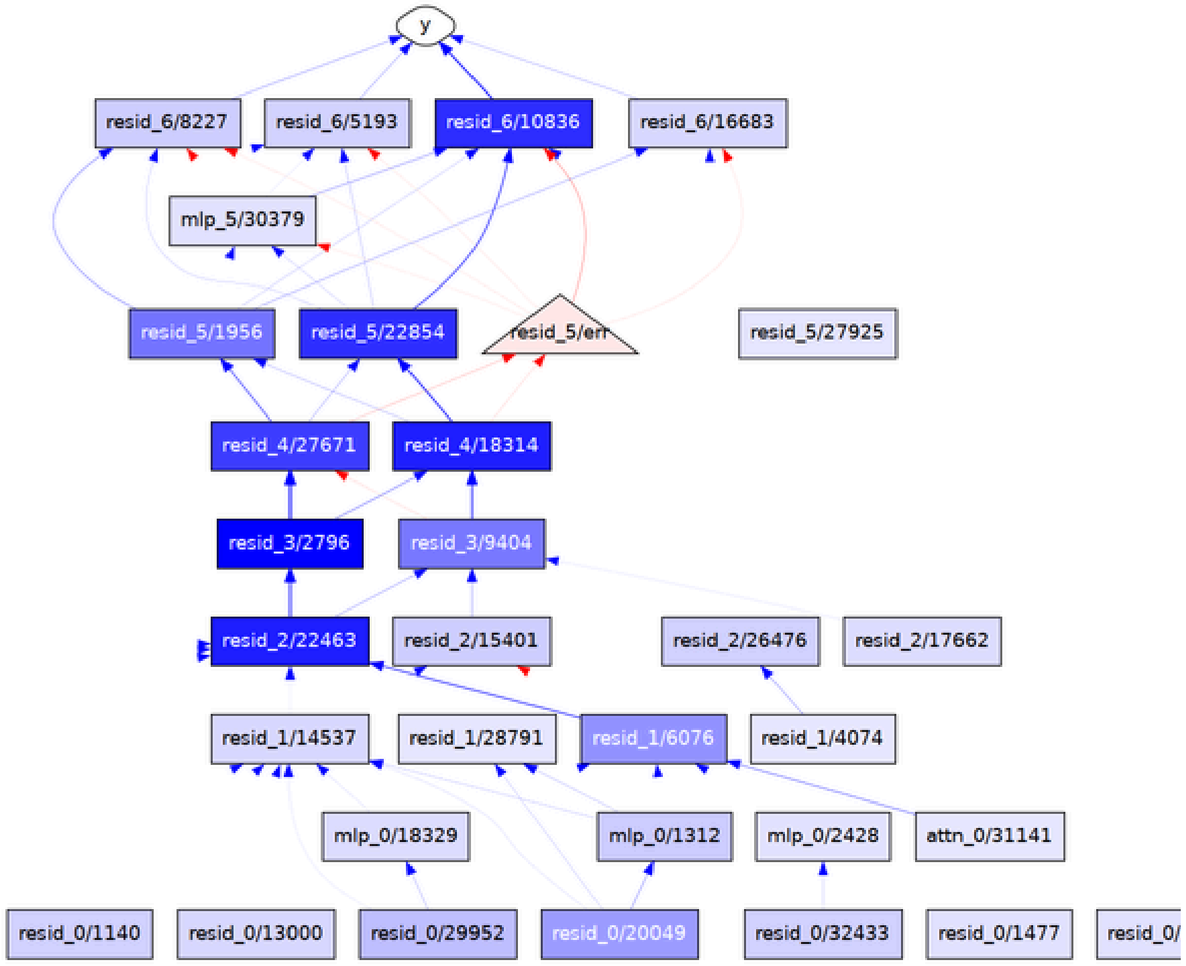


Concept: newline



Concept: comma in the address

# Naming convention:
# {module_type}_{no_of_module}/{SAE feature}



# First contribution - linear version of IE

- *Taylor expansion*

$$\hat{\mathrm{IE}}_{\mathrm{atp}}(m; \mathbf{a}; x_{\mathrm{clean}}, x_{\mathrm{patch}}) = \nabla_{\mathbf{a}} m|_{\mathbf{a}=\mathbf{a}_{\mathrm{clean}}} \left(\mathbf{a}_{\mathrm{patch}} - \mathbf{a}_{\mathrm{clean}}\right)$$

- *Integrated gradients (more precise, alpha from 0 to 1)*

$$\hat{\mathrm{IE}}_{\mathrm{ig}}(m; \mathbf{a}; x_{\mathrm{clean}}, x_{\mathrm{patch}}) = \left(\sum_{\alpha} \nabla_{\mathbf{a}} m|_{\alpha \mathbf{a}_{\mathrm{clean}} + (1-\alpha)\mathbf{a}_{\mathrm{patch}}}\right) (\mathbf{a}_{\mathrm{patch}} - \mathbf{a}_{\mathrm{clean}})$$

**This contribution is not evaluated, we have no runtime benchmark** (instead of that, we have cryptic appendix about fixing this formulas, as they are not good in practise)

# First contribution - circuits on SAE

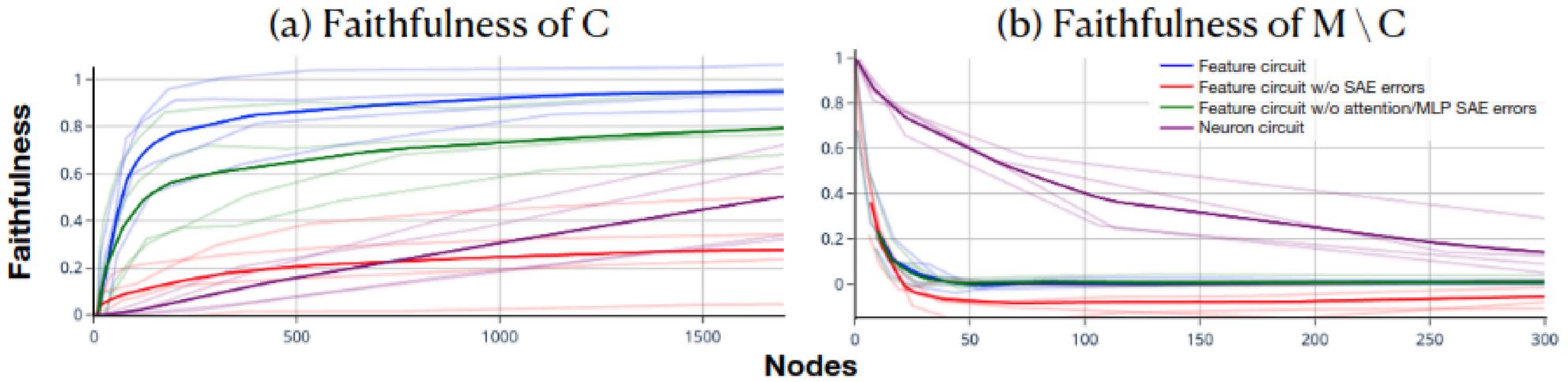
**How to evaluate the quality of circuits?**

- **interpretability - user study**
- **faithfulness**

> **Faithfulness.** Given a circuit $C$ and metric $m$, let $m(C)$ denote the average value of $m$ over $\mathcal{D}$ when running our model with all nodes outside of $C$ mean-ablated, i.e., set to their average value over data from $\mathcal{D}$.[3] We then measure faithfulness as $\frac{m(C)-m(\varnothing)}{m(M)-m(\varnothing)}$, where $\varnothing$ denotes the empty circuit and $M$ denotes the full model. Intuitively, this metric captures the proportion of the model's performance our circuit explains, relative to mean ablating the full model (which represents the "prior" performance of the model when it is given information about the task, but not about specific inputs).

- **Completeness - faithfulness of M\C** (is there any information about the feature outside the circuit? Ideally equal to **0**)

# Second contribution - circuits on SAE



# Second contribution - SHIFT

**Method.** Suppose we are given labeled training data $\mathcal{D} = \{(x_i, y_i)\}$; an LM-based classifier $C$ trained on $\mathcal{D}$; and SAEs for various components of $C$. To perform SHIFT, we:

1. Apply the methods from §3 to compute a feature circuit that explains $C$'s accuracy on inputs $(x, y) \sim \mathcal{D}$ (e.g., using metric $m = -\log C(y|x)$).
2. Manually inspect and evaluate for task-relevancy each feature in the circuit from Step 1.
3. Ablate from $M$ features judged to be task-irrelevant to obtain a classifier $C'$.
4. (Optional) Further fine-tune $C'$ on data from $\mathcal{D}$.

1. find the subgraph of the network that contributes to the classification the most
2. find manually  SAE features that might be harmful
3. remove some features
4. finetune classifier

# Second contribution - comments

1. they use LLM, but the use-case is built upon a linear classifier created on top of the LLM
2. the classifier is trained on down-sampled data that artificially creates bias
3. evaluation is based on data without this bias

| | Accuracy | | |
|---|---|---|---|
| **Method** | ↑Profession | ↓Gender | ↑Worst group |
| Original | 61.9 | 87.4 | 24.4 |
| CBP | 83.3 | 60.1 | 67.7 |
| SHIFT | 88.5 | 54.0 | 76.0 |
| SHIFT + retrain | **93.1** | **52.0** | **89.0** |
| Neuron skyline | 75.5 | 73.2 | 41.5 |
| Feature skyline | 88.5 | 54.3 | 62.9 |
| Oracle | 93.0 | 49.4 | 91.9 |

# Third contribution - automatic feature discovery

We do this in two steps:

1. **Behavior discovery.** We implement variants of the quanta discovery approach from Michaud et al. (2023), which work by clustering contexts based on vectors derived from Pythia-70M activations, gradients or both. Implementation details can be found in App. H.
2. **Circuit discovery.** Given a cluster, we apply the zero-ablation variant of our technique from §3 using dataset $\mathcal{D} = \{(x_i, y_i)\}$, the set of contexts in the cluster together with the next token appearing in The Pile, and metric $m = -\log P(y_i|x_i)$.

# Demo

https://feature-circuits.xyz

# Presenter’s personal opinion

- my personal top 1 chaotic article
- hard to read (each section has a bit different style, the appendix is not really formatted, yet contains important notes)
- novelty of the work is not clear
- some interesting insight was provided but not discussed

# Main concerns

- no real benchmarking, only use-case
- only one small (L)LM, Pythia-70M
- feature circuits were already used; the work only applies them to SAE features (it’s absolutely not clear from the text)
- introduced method, SHIFT, is basically a guideline for feature selection; the role of circuits is not clear, and circuits are too big to be manually analyzed anyway

# Questions?