Evolution of SAE Features Across Layers in LLMs

Daniel Balcells* Independent
Benjamin Lerner* Independent
Michael Oesterle* Independent
Ediz Ucar* Independent

Stefan Heimersheim
Apollo Research

Abstract

Sparse Autoencoders for transformer-based language models are typically defined independently per layer. In this work we analyze statistical relationships between features in adjacent layers to understand how features evolve through a forward pass. We provide a graph visualization interface for features and their most similar next-layer neighbors, and build communities of related features across layers. We find that a considerable amount of features are passed through from a previous layer, some features can be expressed as quasi-boolean combinations of previous features, and some features become more specialized in later layers.

1 Introduction

The goal of Mechanistic Interpretability is to understand how neural networks implement algorithms and store information. The components that we use to define transformers, such as individual attention head neurons (Janiak et al., 2023), MLP neurons (Gurnee et al., 2023), or residual streams, do not seem to be a good basis for analysis, because they can represent multiple functions at once. SAEs (Lee Sharkey, 2022; Cunningham et al., 2023; Bricken et al., 2023) attempt to solve this problem by transforming uninterpretable neural network activations into a high-dimensional, overcomplete basis. These basis dimensions appear more monosemantic than neurons (Bricken et al., 2023).

We typically train and interpret SAEs and their features on each transformer layer independently. While features can be horizontally organised (within a single SAE, Templeton et al., 2024), the features in different layers are not connected "vertically". To understand SAE features, it would be helpful to know their upstream and downstream features. In the extreme (yet as we find, fairly common) case, features are simply duplicated across layers, likely corresponding to directions that are passed through the residual stream, but more complicated relationships are possible.

Related work (Marks et al., 2024) computes causal relationships between SAE features active on a given prompt. Our research instead builds a graph of all SAE features, using cheaper correlational measures to find a more general (not prompt-specific, as advocated for in Nanda (2022)) structure.

Contributions: We create a vertical feature graph, connecting SAE features across adjacent layers based on correlation measures (Figure 2), and provide a web interface to explore the feature graph. Qualitatively, we show that a sizable portion of features in a given layer are passed through from a previous layer (Section 3.1). We find instances in which multiple features in one layer frequently co-activate with a feature in the subsequent layer, in a manner resembling resembling `AND`/`OR` gates (Section 3.2). We investigate subgraphs stretching across multiple layers composed of highly correlated features and show instances of features becoming more specialized in later layers (Section 3.4). We can spot instances in which GPT-3.5 incorrectly labeled features by noticing inconsistent explanations between neighboring features (Appendix L)—we expect this could improve automatic

*Equal contribution. Correspondence to stefan.heimersheim@gmail.com

Workshop on Attributing Model Behavior at Scale at NeurIPS 2024.

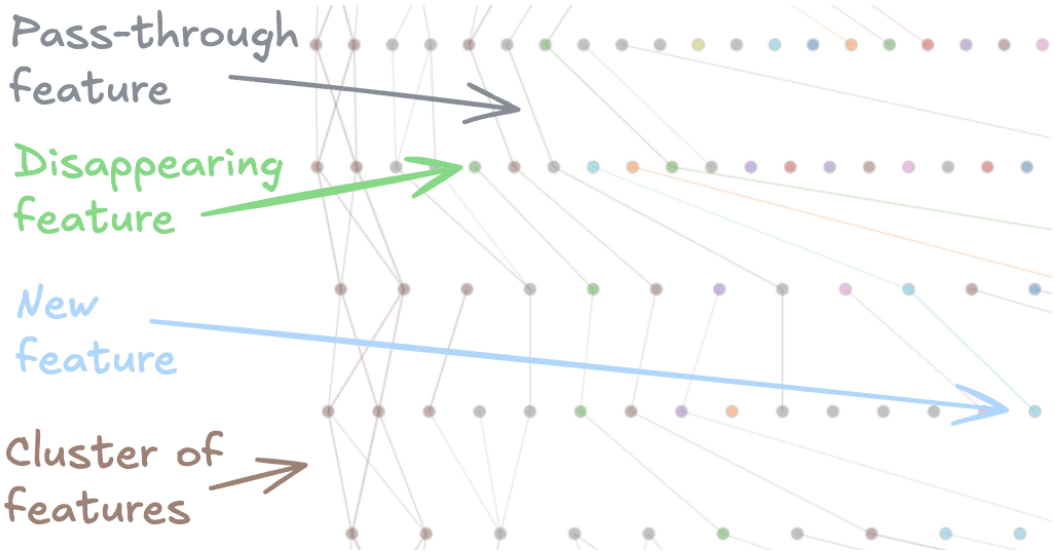


Figure 1: Different motifs we found in our feature graph, active on a single forward pass. Nodes represent SAE features in different layers, earlier layers are at the bottom. (a) "Pass-through" features have high correlation and similar semantic meaning between layers (b) "New" features don't have a counterpart in the preceding layer, we find some that appear to be `AND`/`OR` gates of preceding features. (c) "Disappearing" features don't have a similar feature in the following layer. (d) We find many clusters of related features by running modularity detection algorithms on the full correlation graph, shown here as colors.

interpretability methods. We perform ablation checks and find that our correlational relationships often but not always correspond to causal relationships (Appendix I).

2 Methodology

We leverage GPT-2-small (Radford et al., 2019) and the `res-jb` SAEs (Joseph Bloom, 2024) trained on it. We construct multipartite network graphs with nodes representing SAE features, and edge weights being the similarity measure between two features in different layers. We then conduct network analysis on the resultant graph and explore its structure visually.

We denote SAE features as `L/F` with zero-based indices, e.g., feature 3465 in layer 4 is written as `4/3465`. We leverage Neuronpedia's GPT-3.5 summary of the tokens that most highly activated each feature to annotate SAE features with explanations.

Feature similarity measures: We collect SAE feature activations across all layers over 10M tokens in the Pile (Gao et al., 2020, see Appendix A) and then compute, for all adjacent-layer pairs of features: *Pearson correlation*, *Jaccard similarity* (how often both features fired as a fraction of the number of times at least one fired), *Sufficiency* (how often the downstream feature fired when the upstream feature fired), and *Necessity* (how often the upstream feature fired when the downstream feature fired). See Appendix B for details. We also considered other measures but found them to be less useful in early experiments (Appendix E).

Choosing subsets of nodes and edges for graph visualization: The full similarity graph of GPT-2-small for any given measure consists of $12 \cdot 24576 \approx 300$k nodes and $11 \cdot 24576^2 \approx 6$B edges. We perform post-processing (e.g. modularity calculations, see Section 2) on the full graph, but choose subsets for visualization purposes using either the set of features active on the final token in a given forward pass with a single prompt, or feature subsets based on community clustering algorithm (Section 3.4). We only visualize edges above a similarity threshold.

Detecting communities in SAE feature graphs: We apply community detection algorithms in order to find structure in the graph. The Leiden algorithm (Traag et al., 2019) can find well-connected

communities of nodes - we predict that it would discover groups of semantically similar SAE features. We consider different methods of constructing the feature graph (see Appendix K.1) before running the algorithm in order to yield different partitions of the graph.

Establishing causality between features through ablation: None of the similarity measures above can show that any SAE feature f_k in layer k *causally* contributes to the magnitude of a feature f_{k+1} in layer $k+1$. In order to demonstrate causality, we need to consider counterfactual situations—of all the inputs on which features f_k and f_{k+1} fire in the same forward pass, if f_k's output were set to 0 and everything else were kept constant, what would the average change in f_{k+1}'s output be? In Appendix I we show that the Pearson correlation weakly correlates with the causal effect. We conclude that our feature graph often points out causal relationships, but should primarily be considered correlational.

3 Results

Figure 2: **The community structure within an SAE feature graph**. Nodes represent all the features that were active in the residual stream for a specific prompt of text. The rows of the graph correspond to layers of the transformer such that the bottom row corresponds with the first layer and the top row corresponds to the last layer. The edges between the nodes show the Jaccard similarity between two features > 0.1. The nodes are coloured by the community they were assigned by the Leiden algorithm using the modularity quality function - nodes within a community are semantically similar to one another. For example, the pink community on the far left consists of features related to "Instructions directing on to take action". This graph can be viewed in the feature browser by selecting the `jaccard_leiden_modularity_threshold_0.1_masked_single_23` option in the settings.

3.1 Features being "passed through" multiple layers

We say that a feature in one layer is *passed through* if there is at least one very similar next-layer feature, that it "disappears" if it has no next-layer correspondent, and "appears" if it has no previous-layer correspondent. Figure 3 illustrates the three classes of features per layer, for Pearson correlation similarity with a threshold of $t = 0.95$. We provide a discussion of different threshold choices in Appendix G. Table 1 shows an example of a passed-through feature throughout multiple layers (discovered with the community finding method, see Section 3.4). We see that more than half of the features are appearing and disappearing at each layer, and this fraction increases to 80% for the later layers. This could indicate that earlier layers "build up" features for later use, after which they are dropped from the residual stream.

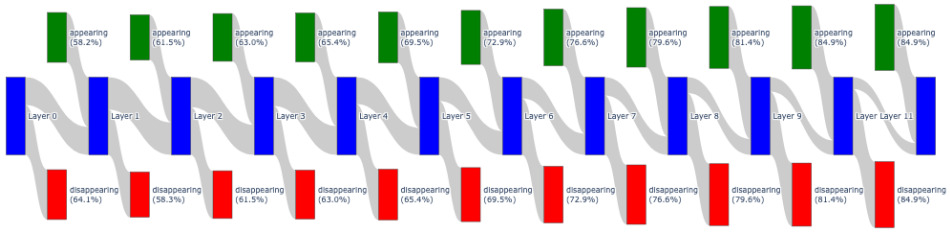


Figure 3: Passed-through/appearing/disappearing SAE features where Pearson ≥ 0.95

Feature	Pearson similarity to next layer feature	Neuronpedia explanation
`0/21891`	0.97	Terms related to panic and anxiety
`1/19235`	0.99	Phrases related to feelings of intense fear or alarm
`2/20272`	0.99	instances of the word "panic"
`3/18762`	0.99	Words related to feelings of fear or distress
`4/17895`	0.98	Words related to a sense of urgency or crisis
`5/21017`	0.96	Words related to the concept of panic

Table 1: Example of a feature being passed through multiple layers. Features denoted as layer/feature.

3.2 Logic Gates

Necessity and sufficiency approximate logical implications if the similarity values are sufficiently close to 1: If, e.g., Necessity$(f_u, f_d) = 0.99$, then it is "99% necessary" that the upstream feature f_u is active for the downstream feature f_d to be active; for two upstream features with this value, it is still "98% necessary" that they both fired. Using these two measures, we can find approximations of `AND` and `OR` relationships between SAE features by searching for downstream features with multiple (but few) upstream neighbours with very high similarity scores. Table 2 shows some features with exactly two neighbours; more examples and a comparison with the other two measures can be found in Appendix H.

3.3 Did features disappear, or did they lack representation in the next layer's SAE?

A feature having no meaningful similarity to any downstream SAE feature could indicate that that feature is no longer present in the downstream layer, or that it is not represented by the SAE but present in its reconstruction error (Marks et al., 2024). To understand the degree to which this happens, we compare the magnitude of an SAE feature i in layer k to the magnitude of layer $k+1$'s error term ($x_{\text{resid}} - x_{\text{resid}} W_{\text{enc}} W_{\text{dec}}$) projected onto $W^k_{\text{dec}}[i]$, as illustrated in Figure 4. We provide a detailed description of this method in Appendix J.

We do not see evidence that disappearing features are present in the next layer's residual stream. This suggests that these features no longer exist downstream, possibly because they split or merge with other features.

Measure	Upstream features	Downstream feature
Necessity (AND)	• 1/9238 (locations and countries in the Baltic region) • 1/12192 (names of places, especially focusing on Estonia and individuals related to legal matters)	2/20513 (mentions of the country Estonia)
Necessity (AND)	• 1/1411 (references to animals, specifically sheep) • 1/13700 (the word "Lamb" preceded or followed by a specific suffix)	2/1218 (mentions of or references to lamb)
Sufficiency (OR)	• 0/303 (references to various organizations or entities with "Life" in their name) • 0/7012 (terms related to MetLife Stadium, Half-Life)	1/17435 (references to the term "Life")
Sufficiency (OR)	• 1/1208 (related to current status or ongoing actions) • 1/6032 (text referring to current states or occurrences)	2/4672 (instances where the term "currently" is used)

Table 2: Feature connections resembling logic gates, where necessity or sufficiency was ≥ 0.999

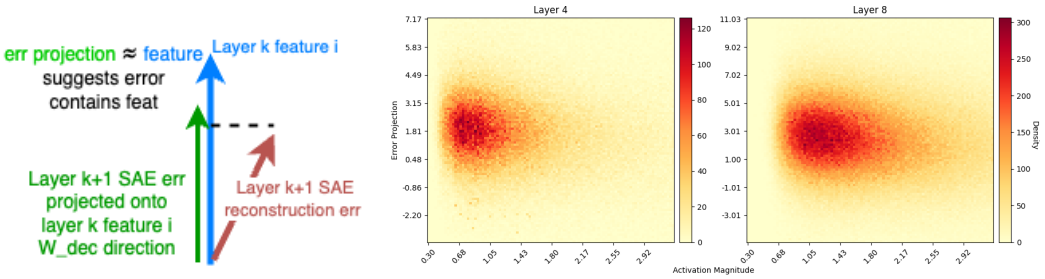


Figure 4: **Left:** Visualizing the recovery of the previous layer's features from the next layer's error term. **Right:** Heatmaps of SAE feature activation vs next-layer error projected onto the feature directions of the previous layer, across many tokens. Only features which "disappeared" (necessity < 0.4 with all next layer features), with an activation $> 0.1\%$ of their max activation are shown.

3.4 Community-detection finds semantically meaningful subgraphs

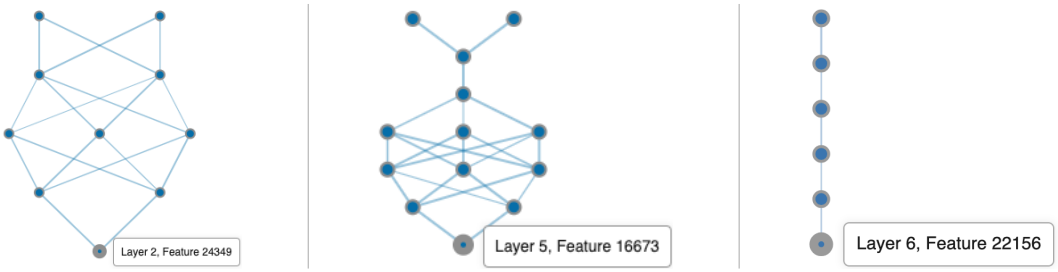


Figure 5: Communities found by applying the Leiden algorithm with a modularity quality function. Intra-layer feature cosine similarity in each community (not used for forming the communities) was measured to be ≥ 0.75. **Left**: Necessity based. 2/24349 detects "evidence" in the context of court cases, causes of economic phenomena, and scientific data. Downstream, 4/8314 specializes in court evidence. **Center**: Necessity based. 5/16673 detects "special" in several contexts, 7/5871 focuses on "Special" in the title of a law enforcement official. **Right**: Jaccard based. 6/22156, and each feature downstream detects the concept of "an important moment".

We used both the Louvain and Leiden community detection algorithms (details in Appendix K.1) to discover subgraphs within the similarity measure graphs. We observe in Figure 5 (more examples in Appendix K.1) a variety of structures within each similarity measures' communities. Simple communities are often long chains of pass-through features, one per layer, sharing many top activating tokens. In more complex communities, features within the same layer have high cosine similarity

and activate on semantically similar tokens. In this way, individual layers in each community have similar properties to the clusters discovered in Bricken et al. (2023). In some communities (Figure 5), we can trace "general" upstream features which detect a token in several contexts to features which appear to "specialize" in detecting that same token in mutually exclusive contexts.

4 Discussion

Our approach measures correlation between features, not causation. This means we do not prove that features interact or that one feature affects the other; in principle, two correlated features could fire via completely unrelated mechanisms. However, it seems more plausible that the downstream feature fires because of the upstream feature (and we see some evidence of that in our ablation results in Appendix I).

The sparsity of SAE feature activations generally results in a low number of active samples, especially since we are interested in co-activations, i.e., tokens where both features of a pair are firing (see Appendix C and particularly Figure 8 for details about the number of co-activating feature pairs). If only a few (say, less than 10) co-activations exist between feature pairs, the identified correlations might be spurious. Run-time constraints do not allow us to use datasets with billions of tokens to avoid this situation. On the other hand, Figure 9 shows that the effect of using 100M instead of 10M tokens has a negligible impact on the similarity matrices.

We chose to focus on four specific similarity measures due to their ease of implementation and widespread use in statistics. However, many other similarity measures exist, and results might differ for other measures.

In addition to our correlation measures, we considered feature geometry: we briefly investigated cosine similarity between features (like Templeton et al. (2024), but for different layers) but found it less useful on its own. A combination of both statistical and geometric measures might further improve the results, but is out of scope for this work.

The choice of hyperparameters—in our case, activation and similarity thresholds—is crucial for managing the wealth of raw data and creating interpretable results and visualisations.

The community detection algorithms used are general purpose. The application of more specialised community-detection algorithms that are more appropriate for sparse multipartite graphs could find more semantically valid communities. A possible strategy for this is the use of a custom quality function for the Leiden algorithm that applies a penalty to the number of nodes used in the same layer. We mostly rely on GPT3.5's summary of the top activating tokens of each SAE feature to understand what it is doing, even though we see in Appendix L that these can be inaccurate.

We do not have a theory describing how feature specialization occurs. More work is needed to determine how a "specific" context is represented mathematically, e.g. comparing the SAE decoder weights for a "general" upstream feature to a "specific" downstream feature and trying to find where the diff lies within the weights of the network.

5 Conclusion

We have shown that SAE features in different layers have clear relationships, and created a graph to represent these relationships. Our visualization is available (here).

We identified communities of features in the graph which have semantically similar interpretations. These include connections resembling `AND` and `OR`-gates, examples of which may help us understand how features are computed in models. Other communities were features being simply passed-through from layer to layer, likely via the residual stream. These features occupy entries of the SAE dictionary without contributing new information. We expect that this insight could be used to create more efficient SAEs across multiple layers. Finally we use the graph to notice connected features with apparently-unrelated explanations, allowing us to spot incorrect (GPT3.5-generated) autointerp explanations. We hope that these correlated features could enrich or verify future autointerp methods.

Author Contributions

Daniel Balcells performed implementation of the batched architecture for similarity calculation, implementation and optimization of Jaccard similarity and decoder cosine similarity measures; design and implementation of the ablation pipeline; design and implementation of the feature browser. **Benjamin Lerner** conducted literature review, implemented ablation pipeline, projected error terms onto feature directions, analyzed communities for structure and meaning, drove the writing and editing of the paper. **Michael Oesterle** designed and implemented the batched architecture for similarity computation, ran experiments to create and compress the matrices, computed statistics (max activations, co-activation of features, distribution of similarity values, pass-through features, ...), implemented the creation and annotation of graphs from prompts, analyzed graphs for structure and meaning, and contributed to writing the paper. **Ediz Ucar** implemented the batched computation of the mutual information similarity measure, conducted an exploratory review of community detection methodologies, implemented algorithm agnostic community detection pipeline, ran experiments using similarity measures to cluster SAE graphs into communities using a broad spectrum of methods, algorithms and parameterisations, and contributed to writing and editing the paper. **Stefan Heimersheim** supervised the project and provided high-level guidance and ideas.

Acknowledgments

This research was supported by the Center for AI Safety Compute Cluster. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

We would like to thank Josh Purtell for giving feedback on the initial idea and providing helpful references to related literature.

References

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models, 2023. URL https://arxiv.org/abs/2309.08600.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The pile: An 800gb dataset of diverse text for language modeling, 2020. URL https://arxiv.org/abs/2101.00027.

Wes Gurnee, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris Bertsimas. Finding neurons in a haystack: Case studies with sparse probing, 2023. URL https://arxiv.org/abs/2305.01610.

Jett Janiak, cmathw, and Stefan Heimersheim. Polysemantic attention head in a 4-layer transformer. https://www.alignmentforum.org/posts/nuJFTS5iiJKT5G5yh/polysemantic-attention-head-in-a-4-layer-transformer, 2023. Accessed on September 10, 2024.

David Chanin Joseph Bloom. Saelens. https://github.com/jbloomAus/SAELens, 2024.

Beren Millidge Lee Sharkey, Dan Braun. [interim research report] taking features out of superposition with sparse autoencoders. https://www.alignmentforum.org/posts/z6QQJbtpkEAX3Aojj/

interim-research-report-taking-features-out-of-superposition, 2022. Accessed on September 10, 2024.

Samuel Marks, Can Rager, Eric J. Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models, 2024. URL https://arxiv.org/abs/2403.19647.

Aaron Mueller. Missed causes and ambiguous effects: Counterfactuals pose challenges for interpreting neural networks, 2024. URL https://arxiv.org/abs/2407.04690.

Neel Nanda. Attribution patching: Activation patching at industrial scale. https://www.neelnanda.io/mechanistic-interpretability/attribution-patching, 2022. Accessed on September 10, 2024.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9, 2019.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter, Chris Olah, and Tom Henighan. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024. URL https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html.

V. A. Traag, L. Waltman, and N. J. van Eck. From louvain to leiden: guaranteeing well-connected communities. *Scientific Reports*, 9(1), March 2019. ISSN 2045-2322. doi: 10.1038/s41598-019-41695-z. URL http://dx.doi.org/10.1038/s41598-019-41695-z.

A Computing feature similarity

Starting from input tokens, we extract the model activations from the `hook_resid_pre` residual stream position of each layer and feed them through the corresponding SAE. The resulting SAE activation vectors are scored pairwise using the similarity measures defined in Section 2.

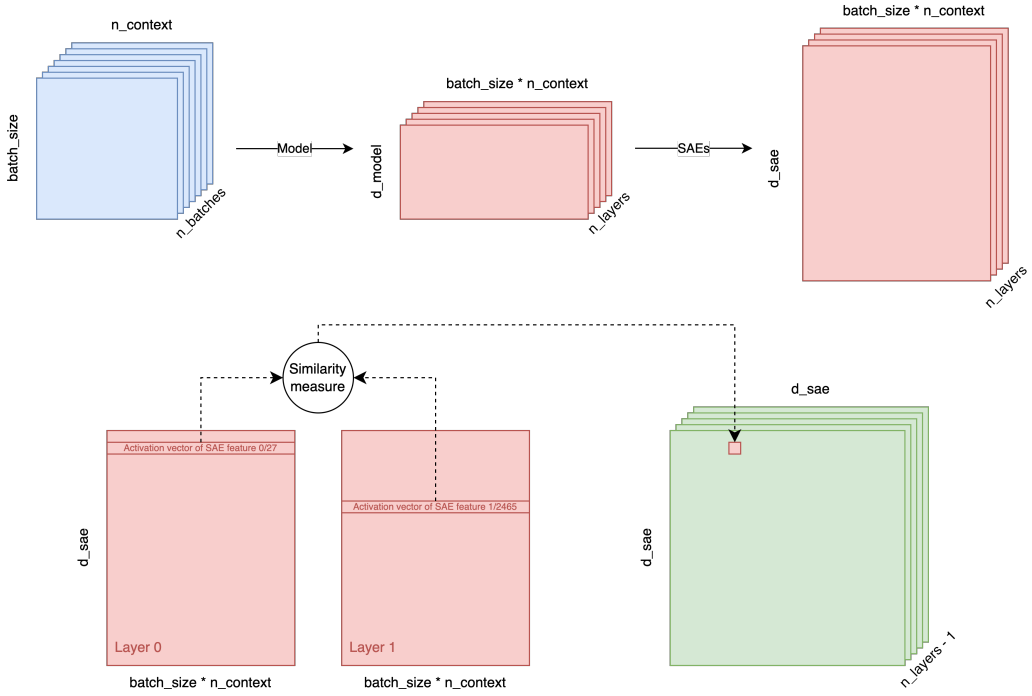


Figure 6: SAE feature similarity computation pipeline from tokens (blue) to model/SAE activations (red) to pairwise feature similarities (green).

B Distribution of maximum activation values per feature

Whether or not an SAE feature is seen as "active" is determined by its (non-negative) activation value. However, the maximum activation values of each SAE over the dataset vary widely (see Figure 7). Therefore, absolute activation thresholds—that is, defining a feature as active if its current activation values exceed a constant value—do not seem to be plausible since this value might never be exceeded by feature f, but almost always for feature f'. Therefore, we use relative activation thresholds throughout the paper, meaning that the threshold value is chosen between 0 and 1, and each feature's activation value is normalised by dividing it by its maximum activation (over the 17.5M token dataset) to determine whether the feature is active. Before computing Jaccard, sufficiency, and necessity we first binarize each feature by defining it as "active" if its activation is at least **20%** of its maximum.

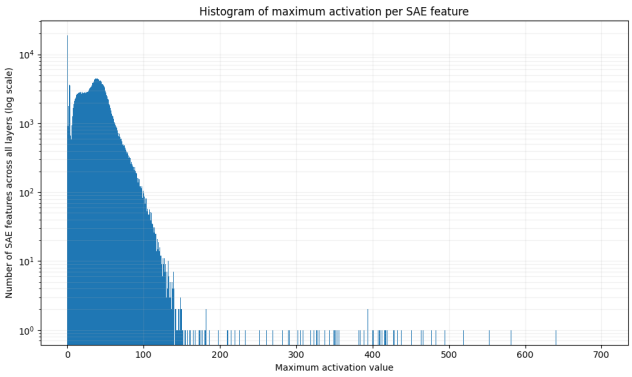


Figure 7: Maximum activation values of all SAE features over the 17.5M token dataset.

C Feature Activation Sampling

Across the 11 pairs of adjacent layers, we measure the percentage of all $\sim$ 6B feature pairs that don't fire together (to avoid spurious correlations, we define "never" as "10 or fewer co-activations") for datasets ranging in size between 500k and 20M tokens. We observe, in accordance with our expectation, that for small datasets, feature sparsity leads to a large proportion of feature pairs not co-activating (see Figure 8). For example, for layers 4-5, using 2M tokens resulted in at least 96% of feature pairs never firing together, compared with only 86% of feature pairs for 10M tokens. Frequent co-activation points to a connection between features, but we need a high number of tokens to obtain a representative sample of co-activations to reliably compute similarities.

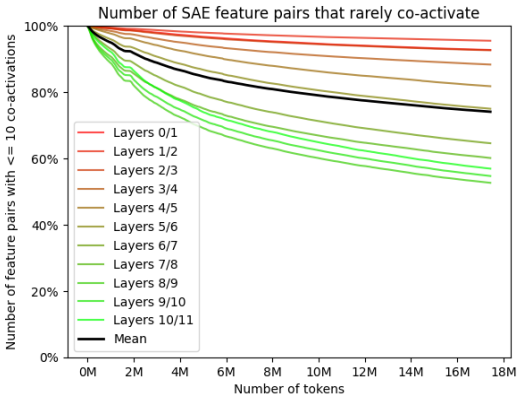
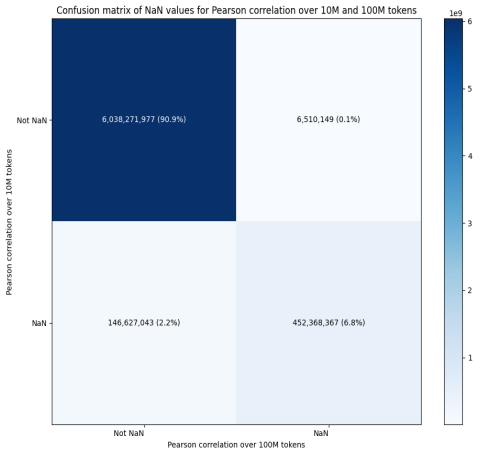
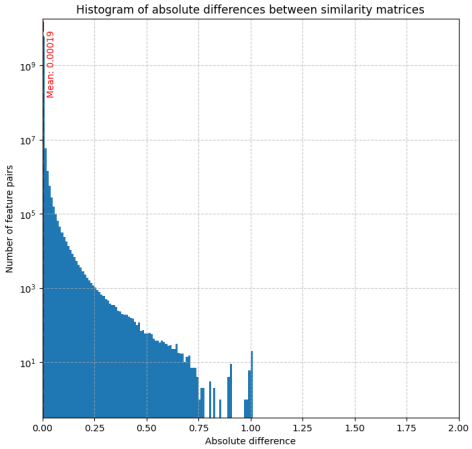


Figure 8: Number of SAE feature pairs from adjacent layers that rarely co-activate.

To verify that the improvement after 10M tokens is acceptably low, we compute the Pearson matrix for 100M tokens and compare its entries to the 10M token version (Figure 9). We find a 97.7% overlap in nan entries—those entries indicate that there was no co-activation of the respective feature pair in the dataset), and the mean absolute difference between corresponding entries is 0.00019.



(a) Confusion matrix of *NaN* entries.



(b) Histogram of absolute differences in similarity values.

Figure 9: Comparing the Pearson similarity matrices for 10M and 100M tokens.

Based on this data, we decide to use 10M tokens for all our similarity measures.

It is not feasible to calculate the similarity matrices after caching all activations since there are 3 trillion activations in total. Instead, for a given pair of layers, we run a batch of 32×128 tokens through the model, cache the SAE features, and compute and accumulate statistics (e.g. counts, sums and sums of squares for Pearson) of the batch. Once all batches have been processed, we finalise the aggregation (e.g. computing Pearson from the final means, variances and standard deviations).

D Distribution of similarity values

The approach described above provides us with a similarity measure for each of the metrics we use, for each pair of consecutive layers, and for each feature pair. In total, this gives us a tensor of shape `(n_metrics, n_layers - 1, n_features, n_features)`$= (4, 11, 24576, 24576)$, which is

$\approx$ 100 GB of raw data. Many similarity values in this matrix are close to zero and, therefore, not likely to describe a strong connection between features. Figure 10 shows the histograms of similarity values, one plot per measure (note the log scale of the y-axis):

Since the similarity values close to zero are not used for downstream processing, we replace all similarity scores below 0.1 with 0.

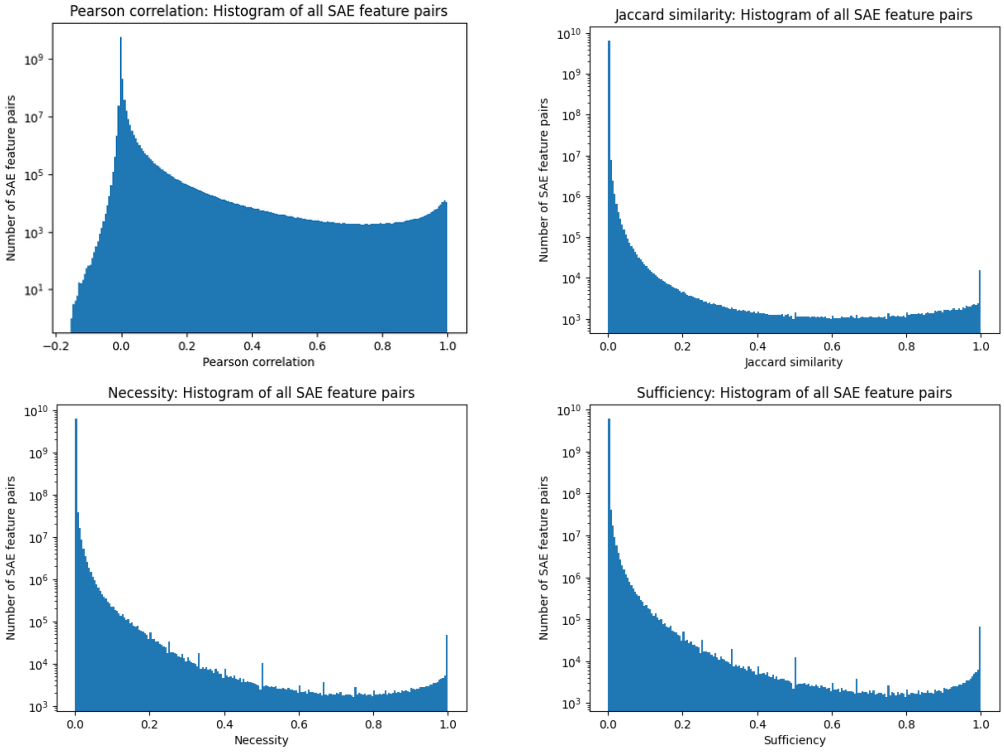
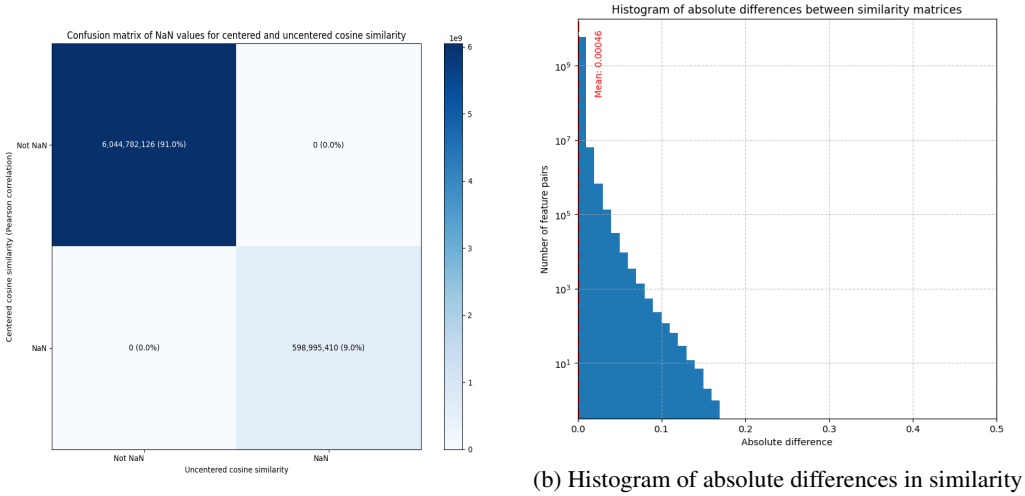


Figure 10: Histograms (for all layer pairs) for the four similarity measures defined in Section 2. Each plot is based on 6B values minus the *NaN* entries produced by non-co-activating feature pairs.

E Additional similarity measures

A fast and easy method to measure the connection between SAE features is to simply compute the pairwise cosine similarity of their decoder weight vectors. The intuition is that, since there are residual connections between the residual streams of subsequent layers, we expect the structure of the RS spaces to be similar. Thus, the directions of SAE features in subsequent layers are more aligned (i.e., their cosine similarity is higher) when their interpretations are more similar. On the other hand, this approach has two weaknesses: First, the internal organisation of the residual streams might slowly (or even disruptively?) change over the course of multiple layers, such that the similarity of feature semantics is not properly represented by the similarity of their directions. Second, this analysis is "static" insofar as it does not take into account the data passed through the model. Early results with this measure were not promising, so we chose not to analyze it further.

One additional statistical measure we considered is an uncentered Pearson correlation $\text{Similarity}(x, y) = 1/N \sum_{i=1}^{N} x_i y_i$ for N samples. This is essentially the "cosine similarity" between the activation sample, but we caution against that name as it has nothing to do with the geometric cosine similarity mentioned above. Figure 11 compares the centered and uncentered Pearson correlation.



(a) Confusion matrix of *NaN* entries.

(b) Histogram of absolute differences in similarity values.

Figure 11: Comparing the similarity matrices for centered (i.e., Pearson) and uncentered cosine similarity.

F Explanation pairs for different similarity values

To build a better understanding about the connection between similarity scores and intuitive "closeness" of textual explanations, Tables 3 to 6 list randomly sampled explanation pairs for a wide range of similarity scores, using all of our measures. Note that the GPT-3.5 explanations are often inaccurate, and checking the top activating tokens on Neuronpedia might provide more insight than just reading the explanations.

Similarity	Upstream feature	Downstream feature
-0.000	3/6743 (phrases related to rebranding or marketing strategies)	4/21451 (words related to being interested or showing interest in something)
0.000	9/849 (positive reviews or feedback on written works)	10/4028 (Phrases related to events leading up to a significant moment)
0.000	2/3681 (proper nouns referring to people or characters)	3/11698 (terms related to jars or containers)
0.100	10/13029 (terms related to laws, rights, and regulations)	11/15564 (words related to general commentary or opinions)
0.136	10/19189 (proper names, potentially female names)	11/19481 (geographic locations, particularly cities, states, and countries)
0.249	2/4867 (words related to negative attributes or actions)	3/11782 (phrases related to the concept of efforts or actions being futile or in vain)
0.208	10/21191 (technology-related terms and concepts)	11/18002 (terms related to digital technologies, tools, and platforms)
0.357	9/8132 (personal pronouns and forms of the verb "to be" associated with self-identification)	10/9550 (phrases related to direct speech and thoughts)
0.415	10/24179 (words related to legal and political entities)	11/18344 (information related to individuals and their actions or controversies surrounding them)
0.433	10/14658 (sources or credits attributions in text)	11/7285 (news sources or citations in a specific format)
0.597	6/23635 (sensitive issues or controversy in a text)	7/13216 (MDA and ADA-related entities)
0.518	6/4867 (conjunctions, specifically the word "and.")	7/16193 (phrases indicating contrast or continuation)
0.659	4/9929 (locations, organizations, and names related to military and government operations)	5/7308 (references to the hockey team "Montreal Canadiens.")
0.646	5/13549 (brands, names, and initials)	6/21248 (names starting with "Nik")
0.757	4/15057 (technical terms and jargon related to various fields or professions)	5/2506 (musical elements, such as instruments and music genres)
0.709	0/3075 (the surname "Mul" with varying numbers following it)	1/23866 (terms related to the city of Varanasi)
0.830	9/3936 (words related to geographical locations and organizations, specifically in the context of British Columbia (B.C.))	10/23868 (locations mentioned along with abbreviated state names)
0.824	8/9666 (long number sequences)	9/12911 (phone numbers)
0.976	10/14867 (proper nouns of people's names)	11/8681 (the name "Don" in various contexts)
0.989	9/1203 (phrases related to various aspects of development, including community development, software development, and personal development)	10/17959 (phrases related to personal or professional growth and progress)

Table 3: Explanation pairs for different values of Pearson

Similarity	Upstream feature	Downstream feature
0.000	6/11361 (words related to finality or conclusion)	7/4252 (references to the iTunes Store)
0.000	3/10208 (words related to different types of crusts)	4/5613 (references to the Islamic State group (ISIS))
0.117	7/22222 (phrases related to crime and violence against specific groups or individuals)	8/5232 (keywords related to legal actions, specifically criminal charges being filed against individuals)
0.132	5/19423 (high-ranking government officials and officials in government positions)	6/23354 (names of political leaders or titles, possibly related to international relations)
0.209	6/19522 (phrases related to inclusion, encompassing everything or everyone)	7/16587 (phrases emphasizing inclusivity and unity)
0.239	7/852 (terms related to action genres, such as action movies and action-packed experiences)	8/18211 (words related to action and physical movement)
0.322	5/16066 (phrases or words that are critical or judgmental in nature)	6/6754 (phrases related to comments or observations made by individuals)
0.363	9/259 (statistics and trends over time)	10/15137 (phrases related to time and history)
0.478	7/6833 (phrases related to time and duration)	8/15460 (durations of time in various units)
0.486	4/5248 (words related to seriousness or urgency)	5/2804 (words related to seriousness and urgency)
0.537	8/19141 (phrases related to legal or governmental mandates)	9/620 (references to political or legal mandates)
0.568	6/19445 (phrases related to improving or reinforcing various aspects, such as control, security, ties, and relations)	7/10759 (phrases related to strengthening or reinforcement)
0.667	2/22884 (references to the name “Sig” in various contexts)	3/14314 (names or terms related to the name “Sig”)
0.622	6/3954 (expressions related to showing approval or disapproval)	7/18469 (mentions of thumbs used metaphorically or literally in various contexts)
0.758	4/23164 (phrases related to monitoring or paying attention to specific things or people)	5/6236 (phrases instructing to pay attention to specific things or events)
0.750	9/21027 (dates in the format of “day number ordinal indicator month year”)	10/21989 (dates expressed in numerical format)
0.898	10/22616 (adjectives that describe the intensity of something)	11/16899 (exaggeratedly positive or negative sentiments)
0.834	7/21991 (references to intellectual property such as patents and trademarks)	8/16537 (phrases related to patents and trademarks)
0.916	4/8882 (terms related to financial derivatives)	5/12890 (terms related to financial derivatives)
0.982	4/23820 (questions or debates revolving around specific topics)	5/22527 (phrases indicating doubt or uncertainty)

Table 4: Explanation pairs for different values of Jaccard

Similarity	Upstream feature	Downstream feature
0.000	8/15979 (mentions of open letters or discussions)	9/9352 (mentions of dense or concentrated things, such as thick fog or thick forests)
0.000	2/2607 (words related to or containing the string "irk")	3/1435 (words related to the concept of defense in various contexts)
0.141	9/6305 (words related to mapping or plotting)	10/17644 (words related to physical body parts and violent actions)
0.160	10/20113 (names of notable individuals and organizations)	11/10398 (references and information related to conflict and military actions in Yemen)
0.227	1/15118 (phrases related to inhibition and inhibitors)	2/19226 (words related to scientific and technical terms like "baseline," "experiment," and "equilibrium.")
0.333	2/8570 (mentions of different technology platforms and devices, specifically iOS and Android)	3/14919 (locations or geographical names)
0.435	10/11796 (references to specific time periods, such as centuries and years)	11/19839 (phrases related to conclusions or summations)
0.498	0/22953 (phrases related to food preparation)	1/4507 (phrases related to preparation or readiness)
0.571	2/10918 (technical terms related to computer programming)	3/2511 (words related to uproar, chaos, and controversy)
0.507	6/8639 (data transfer rates, storage capacities, and other technical specifications)	7/8346 (technical data related to electricity production and consumption)
0.634	0/5889 (terms related to a specific type of technology or material, potentially related to chemical processes)	1/17831 (names of places or countries)
0.642	2/2249 (currency amounts)	3/21826 (dollar amounts mentioned in a sentence)
0.787	5/2675 (instances in which the word "it" is emphasized)	6/10500 (phrases containing "it is")
0.776	8/57 (statements or assertions)	9/14981 (phrases indicating the consequences or implications of a particular situation)
0.826	6/6538 (exact descriptions or comparisons)	7/24347 (specific instances of an exact match in text, such as a repeated phrase or concept)
0.812	6/15935 (adjectives and verbs indicating change or progress)	7/19867 (comparisons or statements about the state or change of things)
0.919	3/22147 (phrases indicating a request or call for action)	4/5847 (phrases indicating an action or decision to pursue a particular course of action)
0.978	2/13173 (the word "wild" and words associated with wildness or untamed nature)	3/10500 (words related to the concept of unpredictability or chaos)

Table 5: Explanation pairs for different values of necessity

Similarity	Upstream feature	Downstream feature
0.000	7/20447 (mentions of the concept of democracy)	8/21210 (superlatives and adjectives suggesting exceptional qualities or rankings)
0.000	10/15422 (mentions of physical cleaning actions, especially involving hands and face)	11/4381 (text related to investigations or examining matters)
0.129	0/932 (mentions of hats)	1/5807 (university names and academic terms)
0.288	9/22155 (phrases related to strong opinions or beliefs)	10/1972 (the word "thought" followed by a number, indicating contemplation or consideration)
0.286	4/3091 (the name "Carmelo" along with various other related words and numbers)	5/23292 (events or entities related to specific "Expos" and person names, such as Neo and Maya)
0.308	0/4369 (names or references to the basketball player Carmelo Anthony)	1/9680 (geographical locations)
0.310	0/761 (occurrences of words related to 'total' or 'summing up')	1/19865 (universities and colleges names)
0.458	1/16591 (words related to administrative or governmental entities, specifically the term "cant" or "cantonment")	2/18040 (mentions or references to the word "Cantonese.")
0.579	2/9833 (mentions of specific brands, particularly HTC)	3/22363 (technology-related terms, particularly related to specific computer brands and components)
0.586	9/11886 (phrases related to academic or professional affiliations, particularly at institutions or organizations)	10/19686 (information related to economic or financial reports and studies)
0.628	8/21608 (mentions of prayer and religious belief, particularly related to Islam)	9/3951 (phrases related to computer programming and code)
0.606	1/23792 (words related to appropriation and misappropriation)	2/4784 (keywords related to legality, propriety, and prudence)
0.746	8/19540 (information and references related to ancient civilizations)	9/21368 (mentions of ancient civilizations, cultures, and history)
0.792	9/19248 (criticisms and negative sentiments in the text)	10/23056 (issues or complexities within processes or environments)
0.846	5/20436 (locations or places, particularly beaches)	6/22365 (references to a specific geographical location, in this case, "Palm Beach.")
0.832	7/878 (references to achieving success or winning in various contexts)	8/4743 (phrases related to success or achievement)
0.961	6/22811 (mentions of geographic locations)	7/2432 (phrases related to states or conditions)
0.963	1/19862 (phrases related to issues or actions involving workers)	2/22786 (references to workers in various contexts, such as productivity, accidents, and disputes)

Table 6: Explanation pairs for different values of sufficiency

G Pass-through features at different thresholds

The number of pass-through vs. appearing and dying features is determined by the similarity threshold defining whether a feature has "passed-through" to the next layer or has no counterpart and has "disappeared". Figure 12 shows the number of high-similarity upstream and downstream neighbours (across all layer pairs) as a function of the similarity threshold. As expected, there are virtually no passed-through features when the threshold is set to 1, and to get a substantial number of features with more than 10 high-similarity neighbours, we have to choose a threshold of 0.2 or lower.

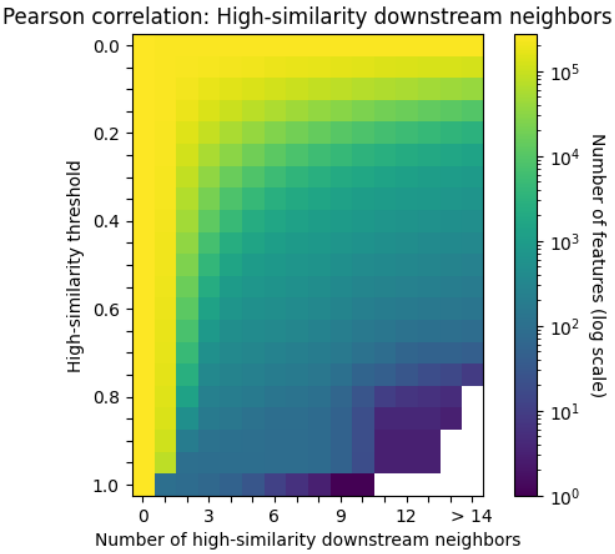


Figure 12: Number of high-similarity downstream neighbours for Pearson as a function of the high-similarity threshold.

To find a suitable threshold, i.e., to calibrate the threshold to separate equivalent features from mere similar ones, we run the following interactive experiment:

1. Start with a threshold of 0.5 and create feature pairs whose similarity is close to this threshold
2. Show random pairs from this set, together with their GPT-3.5 explanations
3. Experimenter decides whether the features of each pair seem to have equivalent meanings
4. Depending on the answer, adjust the threshold to perform a binary search
5. Stop when the threshold range is sufficiently narrow

Running this experiment, we found that ≈ 0.95 is a suitable threshold for Pearson.

H Logic gates

The argument from Section 3.2 that a high value of $n(f_u, f_d)$ means that "f_d approximately implies f_u" equally applies to the *sufficiency* measure, where $s(f_u, f_d)$ means that "f_u approximately implies f_d". By finding a feature f_d with multiple high-similarity upstream neighbours $f_u^{(1)}, ..., f_u^{(n)}$, as done in Table 2, we can systematically find approximate `AND` relationships $f_d = f_u^{(1)} \wedge ... \wedge f_u^{(n)}$, where the meaning (i.e., the explanation) of f_d is the intersection of the meanings of $f_u^{(n)}$ through $f_u^{(n)}$. Similarly, `OR` relationships are found by the sufficiency measure. However, it is important to note that as the number of neighbours increases, the accuracy of these relationships tends to diminish rapidly due to the potential compounding of errors.

The Pearson and Jaccard measures are less clear in what dependency they identify between the features' activation patterns: Pearson extracts the strength of the best linear relationship, while Jaccard measures the overlap between the binarized activation patterns.

In Table 7, we compare our four similarity measures with respect to the informative value of this type of connection: For each measure, we extract feature groups consisting of a single downstream feature and exactly two high-similarity upstream neighbours. We then add explanations to each feature and manually gauge the interpretability of the link between the explanations. Our observations match the theoretical prediction that *necessity* and *sufficiency* and more interpretable in terms of implications than *Pearson* and *Jaccard*.

Measure	**Upstream features**	**Downstream feature**
Necessity	• `1/3696/` (numerical values corresponding to episodes in a series) • `1/9152/` (references to specific episodes)	`2/9633` (references to specific episodes in a TV series)
Necessity	• `0/22068` (words related to redesigning or altering something) • `0/24540` (words related to construction and building refurbishment)	`1/8399` (mentions of renovation or redevelopment of buildings or sites)
Sufficiency	• `3/8555` (legal or financial terms related to seeking some sort of action or outcome) • `3/13751` (the word "sought")	`4/12885` (words related to seeking or looking for something)
Sufficiency	• `0/17409` (terms related to internet browsers - both desktop and mobile) • `0/17478` (mentions of web browsers or actions related to web browsing)	`1/15150` (terms related to web browsers)
Pearson	• `0/19702` (contractions of words to identify emphasis, assertion, or intent in text) • `0/23878` (words related to completion or fulfillment)	`1/12702` (phrases related to inviting and sharing information in a formal context)
Pearson	• `0/17186` (instances of time-related phrases, particularly mentioning the number of days) • `0/21359` (words related to time, specifically focusing on periods of days)	`1/11771` (phrases related to the concept of time durations)
Jaccard	• `0/5985` (concepts related to interpersonal relationships involving exploitation or taking advantage of others) • `0/20876` (phrases from the context repre- senting opinions or perspectives from others)	`1/18220` (instances where actions are performed on or with the involvement of others)
Jaccard	• `0/16222` (political party names, especially the Republican Party) • `0/20665` (phrases related to events, possibly including the word "party")	`1/2979` (references to the Communist Party)

Table 7: Comparison of logic gate identification using all similarity measures, filtering for features with two high-similarity neighbours.

Without defining a quantitative measure, manual inspection indicates that the two "one-directional implication measures", necessity and sufficiency, result in more straightforward semantic connections between features. Of course, this can extended to features with more than two high-similarity upstream neighbours, but manual interpretation becomes harder with more upstream connections.

I Relationship between ablation effect and similarity measure

To compute ablation effects, we ran the following algorithm, following the example set by (Marks et al., 2024) to ensure that the reconstruction loss of the SAE at layer k didn't affect the value of any of the features in layer $k+1$

1. Select a feature f_i in layer k's residual stream SAE.
2. Run a batch of tokens through the model.
3. At layer k, cache activations across the batch for every SAE feature f_j in layer $k+1$, and cache the reconstruction error (difference between original residual stream and residual stream reconstructed from SAE activations).
4. Run the same batch through the model.
5. Intervene in the model at layer k:
    - Encode residual stream into SAE.
    - Ablate f_i, i.e. manually set it to 0.
    - Reconstruct residual stream by decoding the ablated SAE feature vector.
    - Add back the cached reconstruction error stored in step 3.
6. Pass this modified residual stream to layer $k+1$, cache downstream SAE feature values
7. Calculate absolute value of the mean difference in f_j activations for each token with/without ablating f_i

Given that this cannot be run in parallel for multiple upstream features (since ablating two features simultaneously could create interference), we needed to be judicious about which upstream features we chose to ablate. For each similarity measure, for each layer pair, we selected 10 evenly-sized bins of similarity measure scores and randomly 10 sampled feature pairs per each layer pair (e.g. 10 pairs with Pearson between 0.1 and 0.2), and then ablated the upstream feature and recorded the downstream effect.

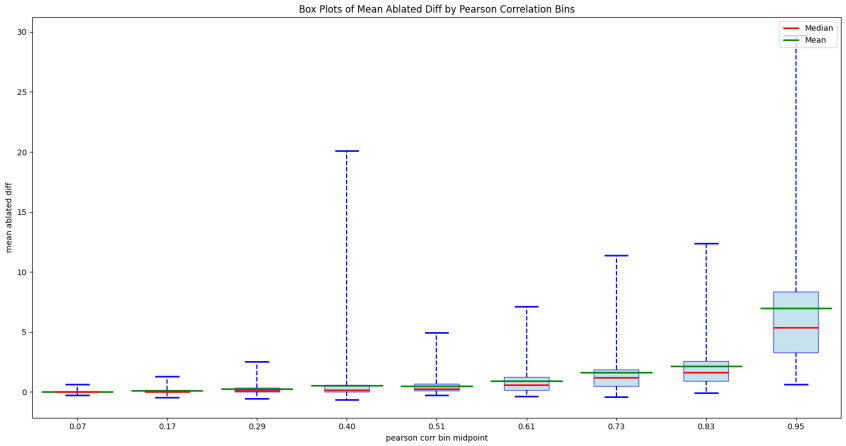


Figure 13

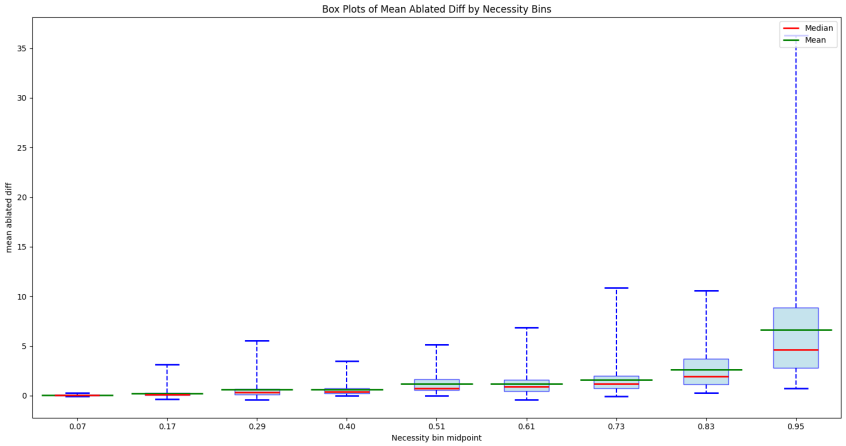


Figure 14

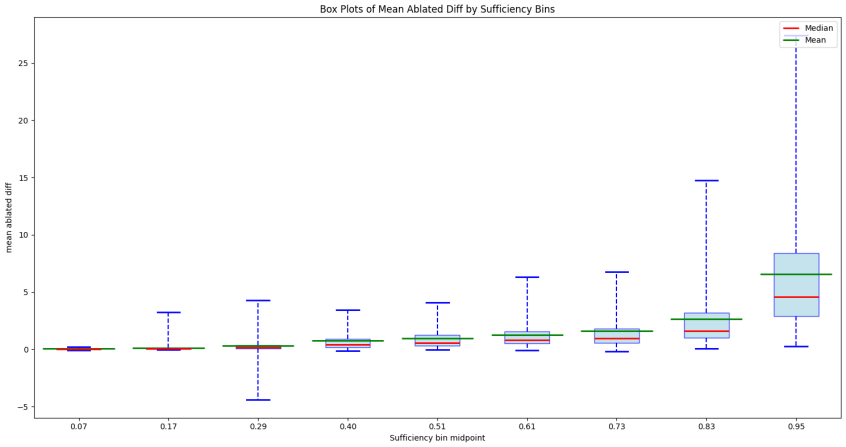


Figure 15

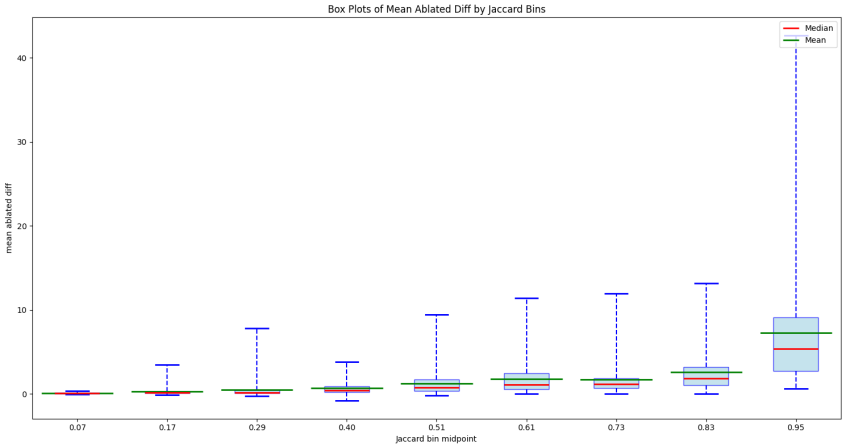


Figure 16

As described above, each similarity measure bin had 10 samples per each of the 11 layer pairs, and the corresponding absolute value of the average change in downstream activation was plotted

with boxplots. The mean ablation effect increases for higher interaction values across all similarity measures, validating that those similarity measures are at least weak indicators of the true causal effect, although Jaccard has the weakest relationship.

A limitation of this approach is, given that we only ablated one feature at a time, any upstream feature redundantly causing a downstream feature's activation alongside a "sibling" upstream feature would appear to have no causation on the downstream feature (because if one is ablated, the sibling would still cause the downstream feature to fire, see Mueller (2024)).

J Error projection method

1. Feed 10M tokens through the model and consider whenever an SAE feature's activation in any layer hit at least 10% of its max activation
2. Calculate the next-layer reconstruction error for that token by
   (a) Recording the residual stream at the next layer
   (b) Encoding and decoding the next layer residual stream with the next layer SAE, the "reconstructed" residual stream
   (c) Computing the difference between the original residual stream and the reconstructed residual stream i.e. where x is the next layer residual stream, computing ε in $x = \hat{x} + \varepsilon(x) = (xW_{enc} + b_{enc})W_{dec} + b_{dec} + \varepsilon(x)$
3. Project that error term back onto the W_{dec} direction of the previous layer's SAE for the feature that activated

Then we plot, for each token, the activation against the error term.

K Community Detection

K.1 Graph Construction

There are myriad hyperparameters to consider when constructing an SAE graph for community detection, such as whether to use weighted or unweighted edges, which quality function to use and the edge weight threshold. One could construct a "full" graph where every node at layer L is connected to every node at layer $L + 1$, with an edge weight equal to the strength of the connection, i.e. the similarity measure. Alternatively, one could construct an unweighted graph with edges added if the similarity measure between them exceeds a threshold value, which therein adds another parameter affecting the resultant graph structure. The choice of graph construction impacts the method of community detection. As an example, both Louvain and Leiden community detection algorithms can be set to either consider edge weights or ignore them.

K.2 Further Examples of Communities

Figure 17: A community found by applying the Louvain algorithm on a graph with edge weights corresponding to Jaccard similarity. This community corresponds to the concept of the C++ programming language. Can be found with name: `jaccard_similarity_louvain_threshold_0.1_size_11_1229` in the feature browser.

Figure 18: A community found by applying the Leiden algorithm with a modularity quality function on a graph with edge weights corresponding to sufficiency similarity. This community corresponds to the concept of operations of various kinds. Can be found with name: `sufficiency_leiden_modularity_threshold_0.1_size_14_61` in the feature browser.

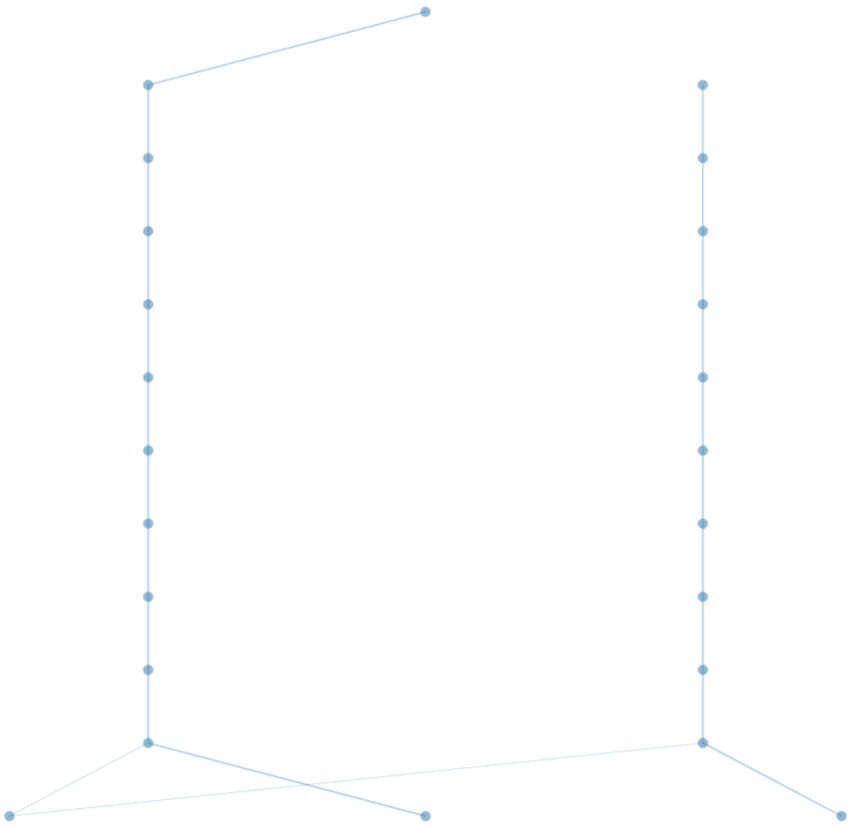


Figure 19: A community found by applying the Louvain algorithm on a graph with edge weights corresponding to Pearson similarity. This community corresponds to two concepts: the name "allen" and the name "adams". Can be found with name: `pearson_louvain_threshold_0.1_size_24_82` in the feature browser.

L Discrepancies between similarity scores and explanation similarity

We find instances in which two features have a high similarity score, but the GPT-3.5 explanations of each are dissimilar. In many of these cases, after looking at the top-activating tokens manually, we find that one of explanations is inaccurate and that they should actually have very similar activations, because both features are detecting the same patterns in the input.

Upstream feature with inaccurate summary	**Downstream feature**	**Pearson score**
`0/60` (words related to padding or cushioning) - it's actually the words "pad" and "pads"	`1/7834` (words related to objects with flat surfaces, often used for various purposes like writing, gaming, or medical applications)	0.9907
`0/134` (words related to a specific geographical location, possibly a country or region) - it's actually "ria"	`1/24445` (the term "ria" in various contexts)	0.992
`3/7661` (names followed by a country or position) - it's actually "aur"	`4/19195` (the word "aur")	0.9976

Table 8: Examples of inaccurate GPT3.5 summaries of features