SPAR: Faithful and Tractable Sparsification of Attribution Graphs for Mechanistic Interpretability

Anonymous EMNLP submission

Abstract

Attribution graphs expose LLM reasoning at the feature level. Still, a single query on a 2-billion-parameter model can produce graphs with 800+ nodes and 13,000+ edges, making interpretation difficult, ad hoc, and cost-prohibitive precisely in the safety-critical domains where it is needed most. We present SPAR (Sparse Pathway Attribution and Reasoning), a post-hoc, model-preserving, local graph sparsification algorithm that operates node-wise, retaining the top-k incoming edges per node proportional to local in-degree rather than applying a uniform global threshold. This preserves circuit integrity at every feature node while aggressively suppressing low-attribution noise. Applied to `Llama-3.2-1B` and `Gemma-2-2B` across CounselBench-Eval, CounselBench-Adv, AdvBench, and AHaBench, SPAR removes up to 43% of edges while maintaining faithfulness within 4% of the original next-token prediction, outperforming global k-NN and random baselines. Perturbation experiments across 200 manipulated queries confirm that SPAR-retained features capture genuine semantic structure rather than surface-level patterns. Code is available at https://anonymous.4open.science/r/spar-E567/README.md

1 Introduction

Trustworthy AI requires understanding not just what a model produces, but which internal features drive that production. Yet polysemanticity (Elhage et al., 2022) ensures that most connections in a neural network conflate unrelated concepts, not every meaningful connection is equally consequential. Even when decomposition into interpretable features succeeds, the resulting feature interaction graph is dense: most connections are present, which ones perform the actual computational work. The operative question is therefore not what any single connection contributes in isolation, but which minimal subgraph of feature interactions preserves the model's computation. That is a problem of circuit sparsification, the construction of a sparse approximation of the feature interaction graph that faithfully reproduces the dense original, and it is the problem this paper addresses.

Attribution graphs, constructed using cross-layer transcoders (CLTs) (Ameisen et al., 2025), make internal feature-level computation visible as weighted directed graphs whose nodes are human-interpretable concepts and whose edges encode linear attribution between them. A single query on a two-billion-parameter model produces over 800 nodes and 13,000 edges. In mental health question answering, where a misattributed feature can shift a model response from supportive to harmful, this density is not just an inconvenience; it is a barrier to safety-critical inspection. Prior efforts have addressed circuit discovery through automated pruning (Conmy et al., 2023; Syed et al., 2024), monosemantic feature decomposition (Lieberum et al., 2024; Ameisen et al., 2025), and post-hoc graph compression (Birardi, 2025; Draye et al., 2025). However, *global* pruning severs pathways at low-degree nodes where any threshold removes all incoming edges; model-modifying approaches invalidate the original computation as a baseline; and dataset-level circuit methods capture average behavior rather than the reasoning pathway of a specific query. What is needed is a method that operates locally at each node, preserves the original model, removes edges rather than features, and produces a faithful sparse graph for each prompt.

This gap motivates three questions. **RQ1:** Can attribution graphs be sparsified post-hoc, without modifying the model, while preserving faithfulness to the original next-token prediction? **RQ2:** Does local edge retention proportional to each node's in-degree preserve circuit integrity better than global pruning? **RQ3:** Do retained features reflect genuine semantic structure, or surface-level token pat-

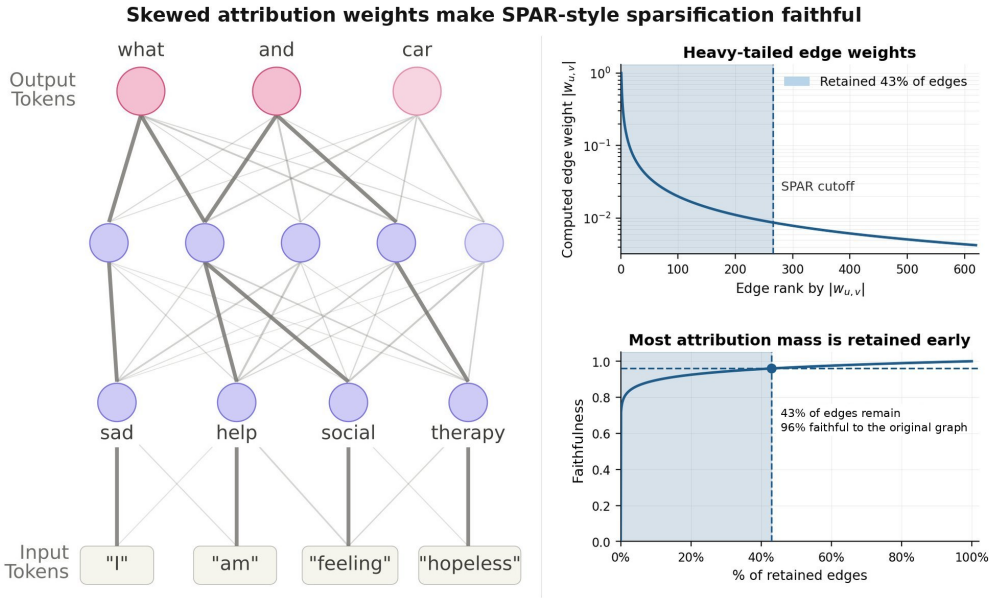


Figure 1: SPAR reduces edges in a circuit while remaining faithful to the original model. Input: "I am feeling hopeless and". Attribution graphs are next-token predictors, so the target is "and". The larger graph contains unhelpful, repetitive, or noisy information. The smaller, darker subgraph holds the edges sufficient to obtain the correct next-token prediction. SPAR's removal of unhelpful information also helps prevent incorrect next-token prediction. $w_{u,v}$ represents the linear attribution node u to node v. u and v are features. Note how SPAR removes the noisy node "car" from the next-token prediction. **SPAR:** Sparse Pathway Attribution and Reasoning.

terns that shift under input variation? We present **SPAR**, a local graph sparsification algorithm that addresses these questions. Our contributions are:

- **A local, post-hoc sparsification algorithm for attribution graphs.** For each node n with in-degree d, SPAR retains the top $k = \lfloor d^e \rfloor$[1] incoming edges by absolute attribution weight, where $e \in (0, 1]$ is a sparsification exponent. This budget respects the local circuit topology without modifying the underlying model.
- **Faithfulness-preserving sparsification at scale.** At e=0.9, SPAR removes 43% of edges, maintaining faithfulness within 4% of the original prediction on Llama-3.2-1B and Gemma-2-2B across CounselBench-Eval, CounselBench-Adv, AHaBench, and AdvBench, outperforming global k-NN and random baselines.
- **Evidence that retained pathways capture semantic structure.** Across 200 knowledge-manipulated queries at three perturbation levels, SPAR-retained features activate consistently under input variation, with concepts such as `medication`, `relationship`, and `therapeutic process` emerging as stable across over 60% of CounselBench-Eval queries.

[1]For simplicity we use top-k here. Details on how top-k is computed are provided in Subsection 2.3.

2 Problem Formulation and Evaluation

2.1 Problem Definition

An attribution graph is a per-prompt record of how a language model reasons. Given an input prompt, a CLT decomposes each layer's activations into monosemantic features and traces, via backward Jacobian tracing, how each feature's activation contributes to downstream features. The result is a weighted directed acyclic graph: nodes are human-interpretable concepts, edges carry signed weights that quantify one feature's causal contribution to another, and the graph terminates at output logit nodes that determine the next-token prediction.

Formally, let M be a transformer-based language model and let $P = (p, c)$ be an input prompt where p is a query and c is associated context. M produces an output distribution $\hat{y} = \text{softmax}(M(P))$ over the vocabulary. We fix the top-k predicted tokens (default $k = 10$) as the target set Y. The attribution graph $\mathcal{AG}$ is a weighted directed graph whose nodes are interpretable features $F_{l,i}$, denoting feature index i at layer l, and whose edges encode linear attribution between features. An edge (u, v) with weight $w_{u,v}$ indicates that feature u con-

tributes $w_{u,v}$ to the pre-activation of feature v in the forward pass. $\mathcal{AG}$ is specific to the pair (P, R) where R is the model's response.

$\mathcal{AG}$ is dense by construction: it retains all attributable paths between features across the full depth of M. This density does not mean that all paths are equally important. Edge weights in $\mathcal{AG}$ are determined by the linearized Jacobian of the model's computation at the point of the forward pass, and their distribution is heavily skewed: a small number of edges carry most of the attribution signal, while the majority carry marginal weight close to zero. The graph is dense because the CLT retains all computed attributions above a minimal threshold, not because the model equally activates all features. *This skewness is what makes principled sparsification possible: signal is concentrated, and low-weight edges can be removed without meaningfully altering the predicted distribution over Y.* The central task is therefore:

> Given a dense attribution graph $\mathcal{AG}$ for a target output set Y, construct a sparse subgraph $\mathcal{SG} \subseteq \mathcal{AG}$ whose retained edges account for the next-token prediction within a defined tolerance, and produces a DAG localizing the information flow responsible for each token in Y.

The fidelity of $\mathcal{SG}$ relative to $\mathcal{AG}$, measured as the closeness of their induced output distributions over Y, is the primary evaluation criterion.

SPAR is a **prompt-level post-inference algorithm**. It operates after the forward pass and graph construction, requires no training, aggregates no information across prompts, and does not modify the model. A new sparse graph is produced for each prompt. This distinguishes SPAR from training-time sparsification methods that alter model weights, dataset-level circuit discovery methods that identify circuits averaged over many inputs, and model-modifying approaches that change the computation under study, shifting the baseline against which faithfulness is measured.

2.2 Intuitive Solution for Sparsification

Consider the query: *"How do you know if you have the right therapist for you? How would you know you've found the right one?"* Its attribution graph contains hundreds of nodes and thousands of edges, many through concepts unrelated to the query. The goal is to identify the edges responsible for the next-token prediction and discard the rest.

A natural starting point is to prune globally: keep the top-k edges by absolute weight. But this treats every node identically. A node with two incoming edges and one with two hundred face the same cutoff. A moderate cutoff severs low-degree pathways entirely while leaving negligible edges at high-degree nodes.

The solution is to give each node its own edge budget, scaled to its degree. This preserves narrow conduits in the circuit while removing low-attribution noise at highly connected aggregation nodes. SPAR implements this principle using a degree-proportional power-law budget, formalized in the following subsection.

2.3 SPAR: Problem Formulation

Let $G = (N, E)$ be the DAG generated for a given prompt, where $|N|$ and $|E|$ denote the total number of nodes and edges, respectively. The sparsified graph is defined as $\mathcal{SG} = (N', E')$, where $N' \subseteq N$ and $E' \subseteq E$. Following the local sparsification principle of Satuluri et al. (2011), for each node $n \in N$ we retain the top k incoming edges ranked by absolute attribution weight, such that:

$$|E'| = \sum_{n \in N} \sum_{i=1}^{k+k'} \mathbf{1} \tag{1}$$

The edge budget $(k + k')$ for node n with in-degree d is computed as: $k = \lfloor d^e \rfloor$, $k' = \lfloor \left(|\delta^{\text{in}}(n)| - k\right)^e \rfloor$, where $e \in (0, 1]$ is the sparsification exponent. Each incoming edge $\in \delta^{in}(n)$, $\forall\, n \in N$ carries a signed weight and is sorted node-wise in descending order of absolute weight to retain the top $(k + k')$ edges per node[2]. The resultant graph $\mathcal{SG}$ satisfies $|E'| << |E|$. Algorithm 1 implements this procedure.

2.4 Variation: Exponent d^e Versus Linear $d \cdot e$

The edge budget in $\mathcal{SG}$ uses a power law: $k = \lfloor d^e \rfloor$. A linear alternative: $k = \lfloor d \cdot e \rfloor$, which retains a fixed fraction e of each node's incoming edges. The distinction is consequential. Under the linear budget, every node loses the same proportion of edges regardless of degree. A node with $d = 2$ retains $\lfloor 2{\cdot}0.9 \rfloor = 1$ edge, losing 50% of its connectivity. A node with $d = 200$ retains 180 edges, losing 10%.

[2] $(k + k')$ = k captures the most important tier of edges at each node using a degree-proportional power-law budget, and k' applies that same budget principle recursively to whatever remains, ensuring that the remainder pool is itself sparsified rather than either fully kept or fully discarded.

Algorithm 1 SPAR: Sparse Pathway Attribution and Reasoning

procedure SPAR($G(V, E),\ e$)
  $E_{\text{keep}} \leftarrow \emptyset$
  Compute in-degree $d_{\text{in}}(v)$ for all $v \in V$
  for each node $v \in V$ **do**
    $k_v \leftarrow \lfloor d_{\text{in}}(v)^e \rfloor$
    $E_{\text{in}}(v) \leftarrow \{(u, v) \in E\}$ ▷ Gather incoming edges
    $k'_v \leftarrow \lfloor (d_{\text{in}}(v) - k_v)^e \rfloor$ ▷ Nested local budget
    Sort $E_{\text{in}}(v)$ by $|w_{u,v}|$ descending
    $S \leftarrow$ top-$(k_v + k'_v)$ edges from $E_{\text{in}}(v)$ ▷ Retain highest-attribution edges
    $E_{\text{keep}} \leftarrow E_{\text{keep}} \cup S$
  end for
  $G' \leftarrow (V,\ E_{\text{keep}})$ ▷ Sparsified graph, satisfying Eq. 1
  return G'
end procedure

The linear budget penalizes low-degree nodes disproportionately, severing pathways at the periphery of the circuit where feature chains are sparse. The power-law budget behaves differently. For small d, $d^e \approx d$, meaning low-degree nodes retain nearly all their edges. For large d, $d^e \ll d$, meaning high-degree nodes are pruned aggressively. At $e = 0.9$ with $d = 2$: $k = \lfloor 2^{0.9} \rfloor = 1$. With $d = 200$: $k = \lfloor 200^{0.9} \rfloor = 108$, retaining 54% of edges. This asymmetry is precisely what is needed: high-degree nodes aggregate information from many features and contain more noise; low-degree nodes are narrow conduits whose removal would sever an entire pathway. The power-law budget respects this asymmetry over the linear budget.

2.5 Evaluation

$\mathcal{SG}$ is only useful if it preserves original graph behavior. We evaluate this along three dimensions: whether the graph still predicts the same token with the same confidence (faithfulness), whether it retains the same total attribution signal (signal retention), whether its output distribution matches the original (distributional similarity), and whether it is minimal, meaning further edge removal immediately degrades it (minimality). We included the full evaluation pipeline algorithm in the Appendix (Algorithm §2).

Faithfulness. We measure faithfulness using the logit difference (LD) between the most likely next token y_{high} and the second most likely token y_{low}, using a variant of work from Mueller et al. (2025):

$$\text{LD}^{(\text{sp})} = \sum_{e:\, * \to y_{\text{high}}} w_e^{(\text{sparse})} - \sum_{e:\, * \to y_{\text{low}}} w_e^{(\text{sparse})}$$

$$\text{LD}^{(\text{orig})} = \sum_{e:\, * \to y_{\text{high}}} w_e^{(\text{orig})} - \sum_{e:\, * \to y_{\text{low}}} w_e^{(\text{orig})}$$

$$\text{Faithfulness} = 1 - \left| \frac{\text{LD}^{(\text{orig})} - \text{LD}^{(\text{sp})}}{\text{LD}^{(\text{orig})}} \right| \quad (2)$$

A value of 1.0 indicates perfect preservation of the original token-level decision boundary. Values below 1.0 reflect deviation in either direction: under-retention, where signal is lost, and the logit difference shrinks, or over-sharpening, where removing noise edges increases the logit difference beyond the original. Both represent a departure from the original computation and are penalized equally. A value below 0 indicates that the deviation exceeds the original signal, corresponding to *catastrophic sparsification*.

Total Signal Retention (TSR). TSR measures the fraction of total edge weight preserved after sparsification:

$$\text{TSR} = \frac{\sum_{e \in E'} w_e}{\sum_{e \in E} w_e} \quad (3)$$

A TSR near 1 confirms SPAR removes low-weight noise edges, not high-attribution signal edges.

Tree Sampling Divergence (TSD). TSD measures whether sparsification preserves the hierarchical clustering structure of the graph. $\phi(A, D)$ is the normalized tree sampling divergence Charpentier and Bonald (2019) quantifying how well the Paris dendrogram D captures the edge structure of adjacency matrix A:
$\text{TSD} = \phi(A_{\text{orig}}, D_{\text{orig}}) - \phi(A_{\text{sp}}, D_{\text{sp}})$
A TSD at or below 0 indicates that sparsification preserves or improves hierarchical coherence.

Minimality: MoRF and LoRF. A minimal circuit is one where every retained edge contributes to the prediction: removing the highest-attribution edges (Most Relevant First, MoRF) should immediately degrade faithfulness, and removing the lowest-attribution edges (Least Relevant First, LoRF) should also immediately degrade faithfulness. We define MoRF and LoRF, inspired by the previous

work by Mumuni and Mumuni (2025). For any sparsified graph $\mathcal{SG}$, MoRF proceeds by removing edges in descending order of $|w_e|$ in chunks and recomputing faithfulness at each step via Eq. 2. LoRF applies the same procedure in ascending order. A graph is minimal if both MoRF and LoRF faithfulness degrade steeply compared to the original graph. Refer to Algorithm §2 in the Appendix that implements both curves.

2.6 Knowledge Manipulation Experiment

Faithfulness metrics confirm that $\mathcal{SG}$ preserves the original next-token prediction, but they do not confirm that retained edges reflect semantic content rather than surface token patterns. A graph tied to token sequences would break under paraphrase; a graph tied to underlying concepts would not. To test this, we construct controlled variants of each query at three manipulation levels: low (single-word substitution), medium (phrase-level rewriting), and high (content-level rewriting), verified within semantic bounds using ParaScore (Shen et al., 2022). A feature is semantically stable if it activates in the SPAR-sparsified graph of at least two of the three manipulated variants, corresponding to a majority-vote criterion across manipulation levels. Table 6 reports features meeting this criterion across CounselBench-Eval queries. Substantial overlap between these stable features and those SPAR retained on the original query constitutes structural evidence that SPAR recovers genuine reasoning pathways.

3 Experimental Setup

Datasets: Evaluating SPAR requires queries that activate a diverse semantically rich set of features across the attribution graph[3]. We evaluate SPAR on four open-ended, safety-critical benchmarks that yield the feature-activation diversity distinguishing domain-stable nodes from prompt-specific noise. **CounselBench-Eval** contains real patient questions from the public forum CounselChat (Bertagnolli, 2020), providing the lexical and semantic diversity needed to identify nodes that activate reliably across the mental health domain rather than within a single prompt type. **CounselBench-Adv** contains 120 adversarial questions authored by 10 licensed mental health professionals, each designed to trigger specific failure modes identified in CounselBench-Eval. These queries stress-test whether SPAR-retained pathways remain faithful under adversarially engineered inputs designed to destabilize model behavior. **AHaBench** contains 5,000 AI safety scenarios with paired chosen and rejected model responses. Its inclusion tests whether SPAR generalizes beyond the mental health domain and whether $\mathcal{SG}$s can distinguish the feature-level differences between safe and unsafe model outputs. **AdvBench** contains 520 prompt-target pairs used to assess LLM safety. This dataset consists of harmful questions spanning 5 security tasks: misinformation, disinformation, toxic content, spam, and sensitive information detection. Prompts from this dataset directly ask the model to perform a malicious task. It enables testing SPAR's performance when the model has been subjected to direct adversarial pressure during inference.

Models and Sparsification. We generate attribution graphs using two LLMs with pretrained CLTs: Llama-3.2-1B and Gemma-2-2B. Both models are intentionally small: their graphs are tractable yet illustrative of the density problem. A single query on Gemma-2-2B produces 859 nodes and 13,832 edges; analyzing thousands of such graphs without sparsification is not feasible. Attribution graphs are generated per prompt using `circuit-tracer` by Hanna et al. (2025). We sparsify each graph at $e \in \{0.8, 0.9, 0.95, 1.0\}$, retaining 33% and 100% of the original graph. Beyond this range, faithfulness drops sharply and KL-divergence grows. Each $\mathcal{SG}$ is evaluated on logit difference, faithfulness ratio, TSR, KL-divergence, TSD, and MoRF/LoRF minimality curves. We repeat the pipeline on perturbed variants to assess how input changes affect retained nodes.

4 Results

Generation Quality: Faithful sparsification is meaningful only if the model still generates coherent responses from the retained graph. We zero out features[4] not retained by $\mathcal{SG}$, run generation, and score the resulting response against the pre-intervention response using BLEURT Sellam et al. (2020), which measures meaning preservation between a candidate and a reference sentence. Table 1 reports BLEURT scores across Gemma-2-2B and Llama-3.2-1B for CounselBench-Eval

[3]Multiple-choice and fact-retrieval benchmarks produce narrow activation patterns idiosyncratic to prompt structure rather than domain content, making it impossible to identify which nodes generalize across queries.

[4]We ignore nodes that are neither in $\mathcal{SG}$ nor in its complement within the intervention domain

Gemma						Llama				
	BLEURT Scores						BLEURT Scores			
e	**0.40**	**0.60**	**0.80**	**0.90**	**0.99**	e	**0.40**	**0.60**	**0.80**	**0.90**
1.00	30	22	14	13	1	**1.00**	10	4	3	2
0.97	31	22	15	14	2	**0.97**	11	3	3	2
0.95	30	21	14	13	1	**0.95**	10	4	4	3
0.90	29	20	13	12	1	**0.90**	11	4	3	3
0.80	34	23	14	13	2	**0.80**	10	2	2	2

(a) CounselBench-Eval

Gemma						Llama				
	BLEURT Scores						BLEURT Scores			
e	**0.40**	**0.60**	**0.80**	**0.90**	**0.99**	e	**0.40**	**0.60**	**0.80**	**0.90**
1.00	96	55	40	36	3	**1.00**	52	19	12	7
0.97	100	60	44	41	4	**0.97**	51	22	14	8
0.95	99	60	42	37	4	**0.95**	53	19	14	8
0.90	100	60	44	39	3	**0.90**	55	20	13	8
0.80	98	59	41	37	3	**0.80**	54	19	13	8

(b) AdvBench

Table 1: Validation results for $\mathcal{SG}$s on CounselBench-Eval and AdvBench. Each entry reports the number of prompts satisfying the corresponding BLEURT score threshold for Gemma and Llama at each value of e. CounselBench-Eval results are out of 95 total prompts, while AdvBench results are out of 520 total prompts. Green cells indicate the best Gemma scores within each BLEURT threshold, while blue cells indicate the best Llama scores within each BLEURT threshold. Aggregating across thresholds, Gemma is maximized at $e = 0.97$ for both datasets; Llama is tied between $e = 0.90$ and $e = 1.00$ on CounselBench-Eval and between $e = 0.80$ and $e = 0.97$ on AdvBench.

Llama-3.2-1B				Gemma-2-2B			
	CounselBench		AdvBench		CounselBench		AdvBench
e	KL_{Eval}	KL_{Adv}	KL	e	KL_{Eval}	KL_{Adv}	KL
1.00	0.000	0.000	0.000	**1.00**	0.000	0.000	0.000
0.95	0.004	0.004	0.002	**0.95**	0.003	0.011	0.000
0.90	**0.019**	**0.019**	**0.010**	**0.90**	**0.038**	**0.046**	**0.003**
0.80	0.087	0.076	0.057	**0.80**	0.354	0.285	0.028

Table 2: KL divergence for Llama-3.2-1B and Gemma-2-2B across CounselBench and AdvBench sparsification levels. Bolded cells indicate that at $e = 0.9$, there is minimal distributional shift between the next-token predictions of the original and sparsified graphs.

and AdvBench, respectively, at sparsification levels $e \in \{0.8, 0.9, 0.95, 0.97, 1.0\}$. Across both models and both datasets, $\mathcal{SG}$s at $e \geq 0.8$ match pre-intervention quality at BLEURT thresholds up to 0.90. At threshold 0.80, Gemma retains 13–15 of 95 prompts on CounselBench-Eval across all $e \geq 0.8$, and Llama retains 2–4, with near-identical counts across exponent values. This plateau confirms a stable operating regime: $\mathcal{SG}$ preserves sufficient feature activation for coherent generation, and increasing retention beyond $e = 0.8$ yields no meaningful gain. Below $e = 0.8$, generation quality degrades (Tables 3 and 4) addressing **RQ1**.

Edge Retention and Faithfulness (RQ1). Tables 3 and 4 report faithfulness and edge retention across all three datasets. At $e = 0.9$, SPAR removes $\approx 43\%$ of edges on CounselBench-Eval (57.2% retained for Llama, 58.7% for Gemma) while maintaining faithfulness within 4% of the original next-token prediction. Notably, at $e = 0.95$, removing 25% of edges produces faithfulness 1.0 for several model-dataset combinations. CounselBench-Adv (Table 3, 4) and AdvBench (Table 3, 4) show the same pattern across both models, confirming that these results generalize across naturalistic, adversarial, and contrastive safety datasets. Table 5 reports Total Signal Retention (TSR) across all conditions. TSR remains above 0.98 for both models at $e = 0.9$, confirming that the removed edges carry negligible attribution weight. Table 2 shows that the KL-divergence remains below 0.05 at $e \geq 0.9$ for Llama across all datasets, and below 0.05 for Gemma on CounselBench-Eval and AdvBench, rising sharply only at $e = 0.8$, where a significant portion of the signal is lost. TSR and KL-divergence confirm that the output distribution is preserved, not merely the top token, which is a stronger faithfulness claim than logit difference alone.

Minimality: MoRF and LoRF (RQ1). The experiments on MoRF and LoRF confirm that $\mathcal{SG}$ is minimal in the sense that every retained edge contributes to the prediction. So, when the highest attribution edges are removed first (MoRF), faithfulness degrades immediately and steeply across all datasets and both models. Similarly, when the lowest-attribution edges are removed first (LoRF), faithfulness degrades more quickly than in the original graph. If the graph contained redundant high-weight edges, MoRF would not degrade faithfulness as sharply. The fact that it does confirms that SPAR has already removed the redundant low-attribution mass, leaving a graph where every retained edge is load-bearing. This result also corroborates the TSR finding: high TSR combined with steep MoRF degradation means the retained edges collectively carry most of the relevant information, and both matter. The MoRF and LoRF curves are in the Appendix as Figures 3 and 4 respectively.

Structural Preservation (RQ2). Table 5 reports

Gemma-2-2B						
	CounselBench-Eval		CounselBench-Adv		AdvBench	
e	ER	F	ER	F	ER	F
1.00	1.000	1.000	1.000	1.000	1.000	1.000
0.95	**0.760**	**0.999**	**0.767**	**1.000**	**0.959**	**0.995**
0.90	**0.587**	**0.969**	**0.597**	**1.000**	0.874	0.983
0.80	0.355	0.940	0.367	0.924	0.657	0.922

Table 3: Faithfulness (F) and edge retention (ER) for Gemma-2-2B across datasets and sparsification levels. Green cells indicate the selected Gemma sparsification setting after excluding $e = 0.97$. Since Gemma BLEURT performance is maximized at $e = 0.97$ in the corresponding validation results, $e = 0.95$ is used here as the nearest reported sparsification level.

Llama-3.2-1B						
	CounselBench-Eval		CounselBench-Adv		AdvBench	
e	ER	F	ER	F	ER	F
1.00	1.000	1.000	1.000	0.999	1.000	1.000
0.95	0.752	1.000	0.753	0.999	0.962	0.985
0.90	**0.572**	**0.974**	**0.573**	**0.975**	**0.881**	**0.959**
0.80	0.333	0.871	0.335	0.881	0.665	0.836

Table 4: Faithfulness (F) and edge retention (ER) for Llama-3.2-1B across datasets and sparsification levels. Blue cells indicate the Llama BLEURT-selected sparsification settings. For CounselBench-Eval, BLEURT performance is tied between $e = 0.90$ and $e = 0.95$; for AdvBench, BLEURT performance is maximized at $e = 0.90$.

TSD across sparsification levels for both models on CounselBench-Eval, CounselBench-Adv, and AdvBench. TSD values are at or below 0 for all $e \geq 0.9$: for example, at $e = 0.9$, TSD reaches -0.018 on Gemma-Eval and -0.017 on Llama-Eval, indicating that the hierarchical clustering structure of $\mathcal{SG}$ is more coherent than that of the original dense graph. This is the expected behavior of a denoising operation: removing low-attribution edges that introduce cross-cluster connections, improves the dendrogram's ability to capture true circuit structure. At $e = 0.8$, TSD reaches -0.046 on Gemma, indicating more aggressive structural reorganization that, while still coherent, begins to deviate from the original topology. These results confirm that SPAR preserves and locally improves the hierarchical organization of feature pathways, directly addressing **RQ2**.

Comparison with Baselines (RQ2). We compare SPAR against k-NN and random sparsification at matched edge counts.[5] At all exponents, SPAR outperforms both baselines on faithfulness at equivalent edge retention. Random sparsification degrades sharply as edges are removed because it has no mechanism for identifying which edges carry a relevant attribution signal. k-NN retains globally high-weight edges regardless of node degree, which severs all incoming edges at low-degree nodes where no individual edge clears the global threshold. SPAR's degree-proportional local budget avoids both failure modes by scaling retention to local connectivity, confirming the advantage of local over global pruning for this task (**RQ2**). We provide the baseline results in Figure 2 in the Appendix.

Semantic Structure of Retained Pathways (RQ3). Table 6 reports CLT features that activate across at least 60% of CounselBench-Eval queries in SPAR-$\mathcal{SG}$s after filtering embeddings and error nodes. The retained features form a coherent layered progression. At Layer 0, the stable features are structural query types: first-person distress openers, yes/no question framing, how-do-I action-seeking openers, and is-it-normal verification-seeking questions. These are the surface forms through which patients initiate help-seeking. At Layer 1, the stable features shift to relational content: acute distress, mid-narrative, and interpersonal conflict. At Layer 4, a general help-seeking opener emerges, and at Layer 5, therapeutic relationship and trust concerns appear. This progression from surface query type at Layer 0 to clinical relational content at Layers 4–5 reflects the semantic architecture of mental health counseling, not the lexical surface of individual prompts. If SPAR were retaining surface-level token patterns, these features would vary across queries rather than stabilizing. The knowledge manipulation experiment reinforces this: across 200 paraphrase-level variants at three manipulation levels, SPAR-retained features activate consistently, with node overlap exceeding 60% of all manipulated graphs. Table 6 and the perturbation results confirm that SPAR recovers genuine domain-level semantic structure, directly addressing **RQ3**.

5 Discussion

The monosemantic nature of the CLT and resulting replacement model allows us to evaluate neu-

[5]No direct comparable baseline exists for monosemantic feature-level attribution graph sparsification; k-NN and random sparsification represent the natural upper bounds for uninformed global edge selection on this substrate. The absence of a stronger baseline reflects the novelty of the task rather than a methodological choice.

Llama-3.2-1B				Gemma-2-2B				Llama-3.2-1B				Gemma-2-2B			
	CounselBench		AdvBench		CounselBench		AdvBench		CounselBench		AdvBench		CounselBench		AdvBench
e	TSR_{Eval}	TSR_{Adv}	TSR	e	TSR_{Eval}	TSR_{Adv}	TSR	e	TSD_{Eval}	TSD_{Adv}	TSD	e	TSD_{Eval}	TSD_{Adv}	TSD
1.00	1.000	1.000	1.000	1.00	1.000	1.000	1.000	1.00	0.0	0.0	0.0	1.00	0.0	0.0	0.0
0.95	1.003	1.006	0.998	0.95	0.990	0.993	0.998	0.95	-0.008	-0.005	-0.0004	0.95	-0.007	-0.007	-0.0004
0.90	**0.998**	**1.003**	**0.993**	**0.90**	**0.985**	**0.985**	**0.992**	**0.90**	**-0.017**	**-0.013**	**-0.0013**	**0.90**	**-0.018**	**-0.019**	**-0.0015**
0.80	0.980	0.988	0.972	0.80	0.955	0.966	0.968	0.80	-0.038	-0.035	-0.0057	0.80	-0.046	-0.046	-0.0070

Table 5: TSR and TSD for Llama-3.2-1B and Gemma-2-2B across CounselBench and AdvBench sparsification levels. Across exponents, signal retention remained high, indicating most of the linear attribution signal was retained. For TSD, at our chosen exponent of 0.9, the values were near 0, meaning the overall structure of the graph was preserved. Note that some rounding errors may cause TSR to go above 1.

L	Human-Interpretable Feature
0	First-person personal distress opener
0	Yes/no and should-I question framing
0	First-person emotional or situational disclosure
0	How/why question-framed help-seeking
0	Mid-narrative personal crisis content (domain-general)
0	Third-party relationship problem framing
0	Mid-narrative distress with self-referential doubt
0	How-do-I action-seeking opener
0	How-can-I coping and control question opener
0	Counselling process and procedural questions
0	Introspective and self-knowledge question opener
0	Is-it-normal verification-seeking question opener
1	Acute distress mid-narrative (concrete crisis events)
1	Interpersonal conflict and relationship rupture
4	General help-seeking opener (broad cross-prompt)
5	Therapeutic relationship and trust concerns

Table 6: Layer-wise progression of semantically stable CLT features across 200 knowledge-manipulated CounselBench-Eval queries. Layer (L) 0 features capture how patients initiate help-seeking through surface query syntax; L 1–5 progressively surface relational crisis content, interpersonal conflict, and therapeutic trust concerns. Because this progression persists across knowledge-manipulation experiment inputs, with over 60% node overlap, it constitutes structural evidence that SPAR retains genuine domain-level semantic organization rather than token patterns. For details on why early layers, we refer readers to the appendix A.6.

ron activations as individual feature attributions to downstream features. We validate our use of d^e in Algorithm 1 by verifying that earlier layers of the attribution graph have more edges than later layers. This ensures we remove proportionally more edges early in the graph. This is a benefit, given that most of the specific query activations occur in the later layers, as demonstrated by our results from Table 6. Other methods produce more compact circuits for next-token prediction, but they must be trained on specific datasets and lack generalizability in their interpretability. Methods that attempt to group edges do not remove noise, so the problem persists. This affects final model outputs and overall causal circuit tracing. We find from our MoRF and LoRF studies that removing additional edges from our graph leads to a sharper decline in faithfulness, suggesting that attribution graphs hold more latent information than neural network edge weights. This work could synthesize both model simplification and post-hoc sparsification to identify a minimal set of causal feature activations in generating certain outputs. Further work could also include optimizing the exponent e values through an exhaustive grid search.

6 Conclusion

The SPAR algorithm provides a mechanism to increase interpretability by removing noisy edges, edges with limited contribution, and inhibitory edges from the attribution graph. A major limitation of attribution graphs was the time required to interpret them. An expert may spend hours analyzing a single graph. After applying SPAR at a threshold of 0.9, the sparsified graph contains 43% fewer edges and removes nodes with $d_{in} = 0$, while retaining faithfulness within 4%. This sparsification provides experts with a smaller subgraph to analyze. It also allows us to identify common node activations across queries in a dataset (Table 6).

SPAR is per-prompt and input-independent, as it is a post-hoc sparsification method, so that it will work for any prompt across domains. Node-level sparsification preserves local attribution weights, preventing the feature loss that global pruning causes. Faithfully reducing models and circuits may not be possible by pruning attention heads or weights, since the modified models may not faithfully represent complex or ambiguous queries. Using SPAR, we can faithfully reduce the density of the attribution graph while retaining the core semantic information necessary to evaluate the safety and trustworthiness of an LLM.

7 Limitations

SPAR operates on attribution graphs, which display only first-order linear edges between features. This means individual edge weights do not capture nonlinear feature interactions directly. However, if SPAR-retained pathways can be organized into inductive, deductive, or abductive reasoning circuits (West et al., 2025; Abdaljalil et al., 2025), they can point toward nonlinear structure even within a linear graph. Confirming that structure requires additional tests: feature ablations, counterfactual prompts, and activation patching to show that circuits interact in context-dependent ways, not just additively. Attribution graphs locate candidate circuits; they do not by themselves prove nonlinear mechanisms. We evaluate on two small models with pretrained CLTs. Extending SPAR to larger models requires the availability of CLT, but the faithfulness-retention trade-off at e=0.9 should hold at larger graph scales on theoretical grounds. SPAR runs in O($|E|$ log d_{max}) time, meaning its cost grows with the number of edges in the attribution graph, not with the number of model parameters. Larger models produce larger graphs, but the property that makes SPAR work, that a small number of edges carry most of the attribution signal while the rest carry near-zero weight, is a known characteristic of trained transformers that holds across model scales. Empirical confirmation on larger attribution graphs remains future work. Finally, the exponent e is chosen empirically; applying SPAR to a new domain requires recalibration against local faithfulness targets.

8 Ethics Statement

Generative AI tools were used for refining overview figures and grammar/spell checking. All data and results figures were not modified by generative AI. Actors intending to cause harm could use SPAR to identify harmful features in the graph and steer the model towards undesirable behavior.

References

Samir Abdaljalil, Hasan Kurban, Khalid Qaraqe, and Erchin Serpedin. 2025. Theorem-of-thought: a multi-agent framework for abductive, deductive, and inductive reasoning in language models. In *Proceedings of the 3rd Workshop on Towards Knowledgeable Foundation Models (KnowFM)*, pages 111–119.

Emmanuel Ameisen, Jack Lindsey, Adam Pearce, Wes Gurnee, Nicholas L. Turner, Brian Chen, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Thomas Henighan, Adam Jermyn, Andy Jones, and 8 others. 2025. Circuit tracing: Revealing computational graphs in language models. *Transformer Circuits Thread*.

Nicolas Bertagnolli. 2020. Counsel chat: Bootstrapping high-quality therapy data.

Giuseppe Birardi. 2025. Automated circuit interpretation via probe prompting. *arXiv preprint arXiv:2511.07002*.

Bertrand Charpentier and Thomas Bonald. 2019. Tree sampling divergence: An information-theoretic metric for hierarchical graph clustering. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI-19)*.

Arthur Conmy, Augustine Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. 2023. Towards automated circuit discovery for mechanistic interpretability. *Advances in Neural Information Processing Systems*, 36:16318–16352.

Florent Draye, Anson Lei, Hsiao-Ru Pan, Ingmar Posner, and Bernhard Schölkopf. 2025. Sparse attention post-training for mechanistic interpretability. *arXiv preprint arXiv:2512.05865*.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. 2022. Toy models of superposition. *Transformer Circuits Thread*.

Javier Ferrando and Elena Voita. 2024. Information flow routes: Automatically interpreting language models at scale. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17432–17445, Miami, Florida, USA. Association for Computational Linguistics.

Michael Hanna, Mateusz Piotrowski, Jack Lindsey, and Emmanuel Ameisen. 2025. circuit-tracer. `https://github.com/decoderesearch/circuit-tracer`. The first two authors contributed equally and are listed alphabetically.

Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, János Kramár, Anca Dragan, Rohin Shah, and Neel Nanda. 2024. Gemma Scope: Open sparse autoencoders everywhere all at once on Gemma 2. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 278–300.

Samuel Marks, Can Rager, Eric J. Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. 2024. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. *arXiv preprint arXiv:2403.19647*. Accepted at ICLR 2025.

Aaron Mueller, Atticus Geiger, Sarah Wiegreffe, Dana Arad, Iván Arcuschin, Adam Belfki, Yik Siu Chan, Jaden Fiotto-Kaufman, Tal Haklay, Michael Hanna, and 1 others. 2025. Mib: A mechanistic interpretability benchmark. *arXiv preprint arXiv:2504.13151*.

Fuseini Mumuni and Alhassan Mumuni. 2025. Explainable artificial intelligence (xai): from inherent explainability to large language models. *arXiv preprint arXiv:2501.09967*.

Neuronpedia. 2025. The circuits research landscape: Results and perspectives. https://www.neuronpedia.org/graph/info. Collaborative report from Anthropic, Decode, EleutherAI, Goodfire AI, and Google DeepMind.

Venu Satuluri, Srinivasan Parthasarathy, and Yiye Ruan. 2011. Local graph sparsification for scalable clustering. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*, SIGMOD '11, page 721–732, New York, NY, USA. Association for Computing Machinery.

Thibault Sellam, Dipanjan Das, and Ankur P Parikh. 2020. Bleurt: Learning robust metrics for text generation. In *Proceedings of ACL*.

Lingfeng Shen, Lemao Liu, Haiyun Jiang, and Shuming Shi. 2022. On the evaluation metrics for paraphrase generation. In *Proceedings of the 2022 conference on empirical methods in natural language processing*, pages 3178–3190.

Aaquib Syed, Can Rager, and Arthur Conmy. 2024. Attribution patching outperforms automated circuit discovery. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 407–416, Miami, Florida, US. Association for Computational Linguistics.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. 2023. Interpretability in the wild: A circuit for indirect object identification in GPT-2 small. In *International Conference on Learning Representations (ICLR)*. Originally arXiv:2211.00593, 2022.

Alva West, Yixuan Weng, Minjun Zhu, Zhen Lin, Zhiyuan Ning, and Yue Zhang. 2025. Abduct, act, predict: Scaffolding causal inference for automated failure attribution in multi-agent systems. *arXiv preprint arXiv:2509.10401*.

Lei Yu, Jingcheng Niu, Zining Zhu, and Gerald Penn. 2024. Functional faithfulness in the wild: Circuit discovery with differentiable computation graph pruning. *arXiv preprint arXiv:2407.03779*.

A Appendix

The SPAR algorithm operates in $O(|E| \log d_{\max})$ time per attribution graph, where $|E|$ is the number of edges and $d_{\max}$ is the maximum node in-degree. For each node v, SPAR sorts its $d_{\text{in}}(v)$ incoming edges by absolute attribution weight in $O(d_{\text{in}}(v) \log d_{\text{in}}(v))$, and summing across all nodes yields $O(|E| \log d_{\max})$, which is effectively linear in $|E|$ since $d_{\max}$ is bounded by graph depth and transcoder sparsity. The dominant cost in the full pipeline is not SPAR itself but attribution graph construction via backward Jacobian tracing, which requires one forward pass and one backward pass per prompt.

Query length is the primary driver of graph size, since longer prompts activate more features across layers, producing more nodes and edges. Ranking the four datasets by estimated query length from longest to shortest: CounselBench-Eval (≈100–200 tokens, naturalistic multi-sentence patient disclosures) > AHaBench (≈150 tokens, truncated in our pipeline) > CounselBench-Adv (≈50–100 tokens, targeted adversarial questions) > AdvBench (≈30–50 tokens, direct single-instruction prompts). This ordering predicts graph density: a CounselBench-Eval query on Gemma-2-2B produces approximately 859 nodes and 13,832 edges, while an AdvBench query produces substantially sparser graphs owing to shorter, less semantically complex inputs. SPAR's $O(|E| \log d_{\max})$ complexity therefore scales most steeply on CounselBench-Eval and most favorably on AdvBench.

On a single NVIDIA H100 GPU (80GB HBM3, 3.35 TB/s memory bandwidth), attribution graph construction for a single prompt on Gemma-2-2B requires approximately 1–2 seconds, dominated by the backward Jacobian pass across all CLT layers. SPAR itself, being a node-wise sort over a graph that fits trivially in memory (≈110 KB per graph at 13,832 edges), executes in under 5 milliseconds and contributes negligibly to wall-clock time. The full pipeline across all four datasets, covering approximately 1,000 prompts in total, completes in roughly 20–40 minutes on a single H100, making SPAR practical for routine safety auditing without specialized infrastructure.

All our datasets are publicly available. The pretrained CLTs for Llama-3.2-1B and Gemma-2-2B are available at Llama-CLT and Gemma-CLT. We use the following configuration from circuit-tracer for graph files generation: `max_n_logits` is set to 30, which limits the number of logits to attributed from; `desired_logit_prob` is set to 0.95, so the attribution is calculated over the minimum number of logits required to reach this probability; `max_feature_nodes` is set to 8192, restricting attribution to only these many feature nodes;

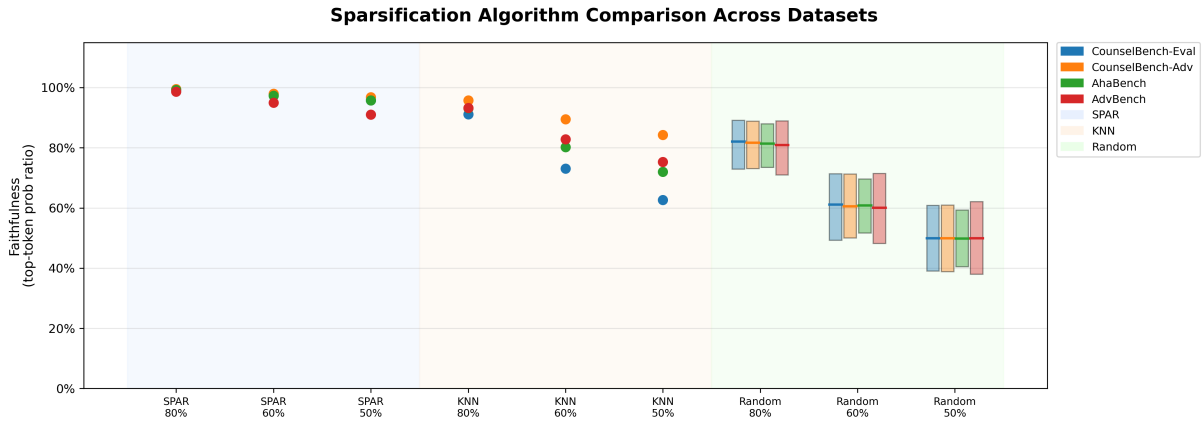


Figure 2: Comparison of SPAR to random and k-NN sparsification faithfulness across all datasets. Random sparsification is over 30 runs. Results indicate that SPAR outperforms random sparsification. We note that each level of sparsification is intended to ensure each sparsifier retains a similar number of edges for comparison (80%, 60%, and 50%). SPAR outperforms random sparsification across all datasets and sparsification levels, and it outperforms all k-NN sparsifiers at varying levels.

`batch_size` is set to 64 during attribution.

A.1 Why SPAR is More Efficient than Brute Force.

The sparsification task is combinatorially hard in its exact form: finding the minimal subgraph $\mathcal{SG} \subseteq \mathcal{AG}$ that preserves faithfulness within tolerance ϵ requires, in the worst case, evaluating all $2^{|E|}$ possible edge subsets, each requiring a forward pass to compute faithfulness. For a graph with $|E| = 13{,}832$ edges, this is computationally intractable. A greedy iterative ablation strategy, analogous to ACDC (Conmy et al., 2023), reduces this to $O(|E|^2 \times F)$, where F is the cost of a single forward pass: at each of $|E|$ removal steps, every remaining edge must be individually ablated and faithfulness re-evaluated, producing approximately 191 million forward-pass-equivalent operations on a typical CounselBench-Eval graph. Global top-k pruning reduces cost further to $O(|E| \log |E|)$ by sorting all edges once, but discards local circuit topology, severing entire pathways at low-degree nodes.

SPAR achieves $O(|E| \log d_{\max})$ with zero forward passes during sparsification. The key efficiency gain comes from two design decisions. First, attribution weights are computed once during graph construction via backward Jacobian tracing and serve as a pre-computed proxy for edge importance; SPAR makes all retention decisions from this signal without querying the model again. Second, the node-wise budget $k = \lfloor d^e \rfloor$ decouples edge decisions across nodes: each node's top-k incoming edges are selected independently via a local sort in $O(d_{\text{in}} \log d_{\text{in}})$, with no combinatorial interaction across nodes to evaluate. Summing across all nodes yields $O(|E| \log d_{\max})$, which on a graph with $d_{\max} \approx 50$ reduces to roughly $O(3.9 \times |E|)$, a near-linear operation. On a single H100 GPU, this means SPAR processes a full CounselBench-Eval graph in under 5 milliseconds, compared to the hours required for expert-guided manual pruning or the days that exhaustive greedy ablation would require at scale.

A.2 Why ACDC is Excluded as a Baseline?

ACDC (Conmy et al., 2023) is excluded on two independent grounds. First, it is computationally intractable at the granularity SPAR operates on. ACDC was designed for circuits over coarse components: attention heads and MLPs, numbering in the hundreds across a model's layers. At that scale, iterative ablation is feasible. CLT-generated attribution graphs contain 800+ nodes and 13,000+ edges per prompt. At this granularity, ACDC's $O(|E|^2 \times F)$ cost produces approximately 191 million forward-pass-equivalent operations per graph, and across the 1,000 prompts in our evaluation pipeline, this extends to 191 billion operations. This is not a practical baseline at the substrate SPAR targets.

Second, and independently, ACDC operates on a fundamentally different object and therefore cannot produce a comparable output. ACDC requires a dataset of contrastive prompt pairs and a predefined task template; it identifies circuits averaged over

that dataset, not per-prompt sparse subgraphs. Its nodes are polysemantic attention heads and MLPs. SPAR operates on a single prompt with no contrastive template, no dataset-level aggregation, and nodes that are monosemantic CLT features. Comparing faithfulness scores between ACDC output and SPAR output would mean comparing objects defined on different substrates, at different granularities, for structurally different purposes. The appropriate baselines are methods that operate on the same object: global top-k (k-NN) and random sparsification of per-prompt CLT attribution graphs, which is precisely what we compare against.

A.3 Related Work

The field of mechanistic interpretability has made rapid progress in identifying the structure of model activations for a given feature or concept, known as a circuit in literature. Early landmark work by Wang et al. manually identified 26 attention heads spanning seven functional categories for the indirect object identification (IOI) task in GPT-2 Small, establishing that transformer computations can be decomposed into interpretable modules Wang et al. (2023). Conmy et al. (2023) automated this process with the ACDC algorithm, which iteratively prunes edges from the computational graph using activation patching. At each node, ACDC removes incoming edges whose ablation changes model outputs by less than a threshold, measured via KL-divergence. ACDC operates at the granularity of attention heads and MLPs, polysemantic components that participate in many unrelated behaviors, and discovers circuits defined over datasets of prompts, not individual inputs. Attribution patching Syed et al. (2024) subsequently demonstrated that a first-order Taylor linear approximation can replicate ACDC's edge-level decisions using only two forward passes and one backward pass, dramatically reducing overall computational cost. More recently, Yu et al. (2024) introduced differentiable computation graph pruning, jointly optimizing weights and edges. These methods share a common property: they identify circuits at the level of coarse model components (heads, MLPs) for aggregate, task-level behavior. Ferrando and Voita (2024) used an attribution-based approach to extract subgraphs representing information flow routes. They do not rely on activation patching. However, they work on the original model components and residual stream computations rather than the sparse representations of the MLP sub-layer.

Other approaches seek to deconstruct polysemantic components into *monosemantic features*. SAEs Lieberum et al. (2024) and CLTs Ameisen et al. (2025) decompose latent representations into interpretable features such that each may correspond to a single concept. Marks et al. (2024) demonstrated that SAE features can serve as nodes in causal circuit graphs, producing sparse feature circuits, but these still involved thousands of nodes and hundreds of thousands of edges. Ameisen et al. (2025) combined CLTs with backward Jacobian tracing to produce *attribution graphs*, complete, per-prompt, feature-level circuit maps that show how every interpretable feature contributes to the next-token prediction through weighted directed edges. This represented the highest-resolution view of model computation to date, but it also made the density problem acute.

Parallel efforts have attempted to address this density, but each operates on different levels of the model than the level SPAR targets:

Model-level sparsification. Draye et al. (2025) introduced sparse attention post-training, applying sparsity regularization to attention patterns as a post-training intervention. Their method reorganizes information flow within the model itself, reducing edges between components by up to two orders of magnitude while retaining essential algorithmic behaviors. Critically, this approach modifies the model's attention patterns. It produces a sparser model whose circuits are then easier to trace, but it changes the computation under study.

Semantic compression of attribution graphs. Birardi (2025) proposed probe prompting, which transforms dense attribution graphs into compact subgraphs by grouping similar features (Therapist, Psychologist, Doctor) into a single node (Mental health professional jobs), called a supernode. This reduces interpretation time from approximately two hours to ten-twenty minutes of semi-automated processing. Probe prompting operates on the same object as SPAR, post-hoc attribution graphs, but focuses on the density problem through semantic grouping of nodes.

Default graph pruning. The Neuronpedia (2025) circuits research landscape report notes that attribution graphs are typically pruned prior to analysis using the threshold-based algorithm described by Ameisen et al. (2025), and that achieving the right level of graph sparsity requires balancing

transcoder-level sparsity with graph-level pruning.

We propose a post-hoc, model-preserving, edge-level sparsification of per-prompt attribution graphs that operates locally at each feature node. ACDC and its descendants prune coarse-grained circuits over datasets. Birardi (2025) modify the model. Probe prompting groups nodes but does not remove edges. Ameisen et al. (2025)'s default pruning applies a global threshold.

A.4 Algorithms

Algorithm 2 Evaluation: Faithfulness, MoRF, and LoRF

procedure EVALUATE($G, \mathcal{SG}, Y, \textit{steps}$)
  Compute $\mathrm{LD}^{(\mathrm{sp})}$ and $\mathrm{LD}^{(\mathrm{orig})}$ over edges into y_{high} and y_{low}
  $r_{\text{faith}} \leftarrow \mathrm{LD}^{(\mathrm{sp})} \;/\; \mathrm{LD}^{(\mathrm{orig})}$ ▷ Eq. 2
  Compute $P_{\text{sparse}}(t_{10})$ and $P_{\text{orig}}(t_{10})$
  $r_{\text{prob}} \leftarrow P_{\text{sparse}}(t_{10}) \;/\; P_{\text{orig}}(t_{10})$
  ▷ **TSR** (Eq. 3)
  $\mathrm{TSR} \leftarrow \sum_{e \in E'} w_e \;/\; \sum_{e \in E} w_e$
  ▷ **MoRF**: remove highest-attribution edges first
  Sort E' by $|w_e|$ descending; $\textit{chunkSize} \leftarrow \max(1, |E'|/\textit{steps})$
  for step $s = 0$ to *steps* **do**
    Remove top $s \cdot \textit{chunkSize}$ edges from E'
    Compute faithfulness on remaining edges via Eq. 2
    Record (edges remaining, faithfulness)
  end for
  ▷ **LoRF**: remove lowest-attribution edges first
  Sort E' by $|w_e|$ ascending; repeat loop above
  return r_{faith}, r_{prob}, TSR, MoRF curve, LoRF curve
end procedure

Algorithm 3 Faithfulness Calculation

Require: $G(V, E)$: The sparsified attribution graph; L_{corr}: The target (correct) label; L_{incorr}: The contrastive (incorrect) label.

Ensure: diff: The faithfulness score representing the logit difference.

procedure FAITHFULNESS($G(V, E), correctLabel, incorrectLabel$)
  $y_{corr} \leftarrow 0$
  $y_{incorr} \leftarrow 0$
  $E_{corr} \leftarrow \{(u, v) \in E \mid v = \mathrm{Node}(L_{corr})\}$ ▷ Edges to correct output
  $E_{incorr} \leftarrow \{(u, v) \in E \mid v = \mathrm{Node}(L_{incorr})\}$ ▷ Edges to incorrect output
  for each edge $(u, v) \in E_{corr}$ **do**
    $y_{corr} \leftarrow y_{corr} + w_{uv}$
  end for
  for each edge $(u, v) \in E_{incorr}$ **do**
    $y_{incorr} \leftarrow y_{incorr} + w_{uv}$
  end for
  diff $\leftarrow y_{corr} - y_{incorr}$
  return diff
end procedure

Algorithm 4 MoRF

procedure MoRF($G(V, E), steps$)
  $E_{keep} \leftarrow set()$
  $chunkSize = max(1, \frac{|E|}{steps})$
  sort edge values in decreasing order $\forall\, e \in E$
  for each step s from 0 to $steps$ **do**
    $idx \leftarrow s \cdot chunkSize$
    $E_{keep} \leftarrow \{e_i \in E \mid i > idx\}$ ▷ Remove the idx most relevant edges
    Calculate faithfulness for E_{keep}
    $E_{rem} \leftarrow \{e_i \in E \mid i \leq idx\}$ ▷ Save removed edges
    $w_{rem} \leftarrow \sum_{e \in E_{rem}} |w_e|$
  end for
  return Faithfulness, w_{rem}, and idx
end procedure

Note: For LoRF, we reverse the sorting order and sort edges based on the least relevant edges.

AHaBench		
e	**Llama-3.2-1B**	**Gemma-2-2B**
1.00	0.000	0.000
0.95	0.001	0.026
0.90	**0.007**	**0.089**
0.80	0.046	0.595

Table 7: KL divergence for Llama-3.2-1B and Gemma-2-2B on AHaBench across sparsification levels.

A.5 Additional Results

In this subsection, we demonstrate results on MoRF and LoRF in Figures 3 and 4 respectively. Table 6 represents the common CLT Features identified across manipulated CounselBench-Eval prompts. Tables 7, 8, 9, 10, 11, and 12 represent observations on the AHaBench dataset.

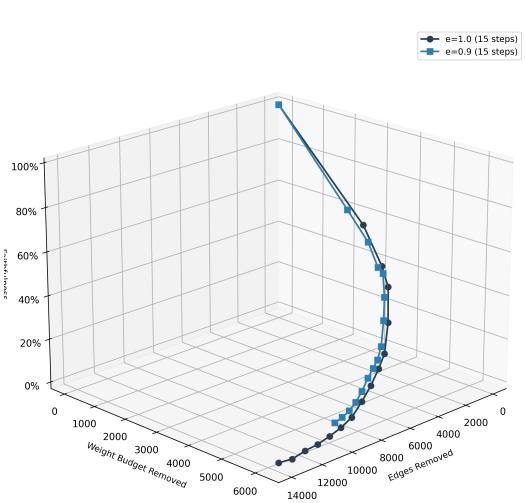
Llama				
	BLEURT Scores			
e	**0.40**	**0.60**	**0.80**	**0.90**
1.00	6	1	1	1
0.97	4	1	1	1
0.95	5	1	1	1
0.90	7	1	1	1
0.80	8	1	1	1

Table 8: Validation results for $\mathcal{SG}$s on the AHaBench dataset. Each entry reports the number of prompts, out of 200 total prompts, that satisfy the corresponding BLEURT score threshold for Llama at each value of e. Blue cells indicate the best Llama scores within each BLEURT threshold. Aggregating across thresholds, Llama is tied between $e = 0.80$ and $e = 1.00$.

A.6 Why Early Layers in Table 6?

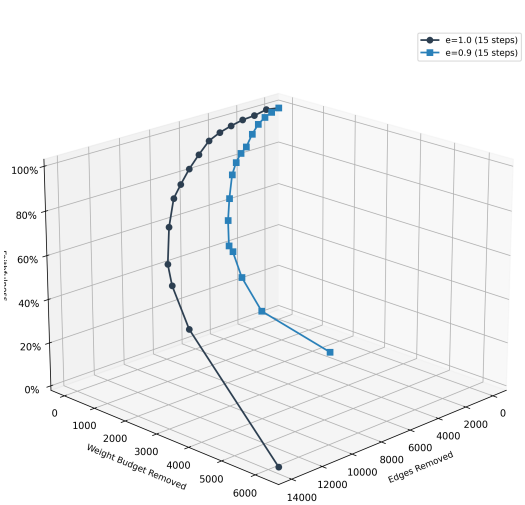
`Llama-3.2-1B` has 16 layers. `Gemma-2-2B` has 26. Stable features appear only through Layer 5. That is not a ceiling imposed by SPAR. It is a window into how these models actually partition computational work across depth.

Consider what the CLT is decomposing at each layer. In early layers, the monosemantic features that emerge correspond to broad semantic categories: communicative intent, discourse structure, and domain membership. These are the features that must be stable across paraphrases precisely because they are responding to what the speaker is doing rather than what words they chose to do it with. A first-person distress opener is a first-person distress opener, whether the patient says "I don't know how to cope" or "I'm struggling to manage."



Edges Rem.	Edges Left	Faith 1.0	Faith 0.9
0	13830	100.0%	100.0%
922	12908	59.2%	62.6%
1844	11986	46.9%	53.0%
2766	11064	42.3%	45.2%
3688	10142	30.2%	45.7%
4610	9220	19.7%	37.5%
5532	8298	16.3%	29.4%
6454	7376	11.8%	19.7%
7376	6454	7.7%	15.8%
8298	5532	3.6%	13.8%
9220	4610	1.9%	11.4%
10142	3688	0.8%	7.2%
11064	2766	0.1%	4.1%
11986	1844	-0.0%	2.0%
12908	922	-1.0%	0.8%
13830	0	0.0%	0.0%

Figure 3: Most Relevant First (MoRF) on our example query from section 2.2 "How do you know if you have the right therapist for you? How would you know you've found the right one?". Results indicate that MoRF on e=0.9 faithfulness declines faster than e=1.0



Edges Rem.	Edges Left	Faith 1.0	Faith 0.9
0	13830	100.0%	100.0%
922	12908	101.0%	99.2%
1844	11986	100.0%	98.0%
2766	11064	99.9%	96.1%
3688	10142	99.2%	92.8%
4610	9220	98.1%	88.3%
5532	8298	96.4%	86.9%
6454	7376	92.3%	84.2%
7376	6454	88.2%	80.3%
8298	5532	83.7%	71.3%
9220	4610	80.3%	63.4%
10142	3688	70.6%	54.3%
11064	2766	57.7%	54.8%
11986	1844	53.1%	47.0%
12908	922	40.8%	37.4%
13830	0	0.0%	32.8%

Figure 4: Least Relevant First (LoRF) on our example query from section 2.2 "How do you know if you have the right therapist for you? How would you know you've found the right one?". Results indicate that LoRF on e=0.9 faithfulness declines much faster than e=1.0. This indicates that the least relevant edges in $e = 0.9$ are necessary, as removing them causes a sharper decline in faithfulness.

AHaBench		
e	**Llama-3.2-1B**	**Gemma-2-2B**
1.00	1.000	1.000
0.95	1.000	0.996
0.90	**0.998**	**0.991**
0.80	0.990	0.979

Table 9: Total signal retention (TSR) for Llama-3.2-1B and Gemma-2-2B on AHaBench across sparsification levels. At $e = 0.9$, TSR remains close to 1 for both models, indicating that the majority of attribution is kept in $\mathcal{SG}$.

AHaBench		
e	**Llama-3.2-1B**	**Gemma-2-2B**
1.00	0.0	0.0
0.95	-0.0006	-0.0006
0.90	-0.0019	-0.0020
0.80	-0.0080	0.0098

Table 10: Tree Sampling Divergence (TSD) for Llama-3.2-1B and Gemma-2-2B on AHaBench across sparsification levels.

The concept is invariant to the surface form. That invariance is exactly why these features survive the 60% stability threshold.

Later layers are doing something categorically different. They are computing the next-token prediction, which is necessarily sensitive to the exact context. By Layer 10 or Layer 15, the model is no longer asking what domain this query belongs to; it is asking what word comes next, given everything that preceded it. That computation is prompt-specific by design. Features at those depths will not activate consistently across paraphrase variants because they are not supposed to. Their specificity is their function.

Our study actually confirms this directly in Section 5, noting that most specific query activations occur in later layers. The implication is profound: SPAR's sparsification, which removes proportionally more edges from early layers, where degrees are higher, selectively denoises precisely the layers where domain-stable semantic structure resides. What survives is the part of the circuit that encodes the clinical meaning of the query. What is removed is the part that encodes its token-level surface realization.

The gap between Layer 1 and Layer 4 is equally interesting, and the paper does not address it. Layers 2 and 3 produce no stable cross-prompt features.

Gemma-2-2B on AHaBench		
e	ER	F
1.00	1.000	1.000
0.95	**0.748**	**0.973**
0.90	0.567	0.954
0.80	0.332	0.911

Table 11: Faithfulness (F) and edge retention (ER) for Gemma-2-2B on AHaBench across sparsification levels.

Llama-3.2-1B on AHaBench		
e	ER	F
1.00	1.000	0.998
0.95	0.750	0.996
0.90	**0.570**	**0.969**
0.80	0.333	0.865

Table 12: Faithfulness (F) and edge retention (ER) for Llama-3.2-1B on AHaBench across sparsification levels.

This suggests that intermediate layers are performing transformational work, routing and composing features, which is inherently input-dependent and therefore unstable under manipulation. They are the mechanism, not the meaning.

To sum it up, early layers recognize what kind of question is being asked, and that recognition does not change when you rephrase the question. Later layers are busy figuring out what word comes next, which changes every time the input changes. SPAR's stability criterion naturally finds the early layers because that is where the meaning lives.