Decision Path Tracing for Causal Analysis in Transformers

Won Jo
KAIST
clockjw@kaist.ac.kr

Dahee Kwon
KAIST
daheekwon@kaist.ac.kr

Jongeun Baek
KAIST
jongeun.baek@kaist.ac.kr

Cheongwoong Kang
KAIST, INEEJI
cw.kang@kaist.ac.kr

Jaesik Choi
KAIST, INEEJI
jaesik.choi@kaist.ac.kr

Abstract

Investigating the causal structures of transformers often entails a trade-off between the granularity of the explanation and computational overhead. Existing approaches either bypass the full internal chain, potentially missing critical decision-relevant links, or struggle with the combinatorial explosion of path enumeration. To address these challenges, we propose an efficient automated path-tracing method that ensures both causal reliability and algorithmic feasibility. Through extensive verification, we find that (i) the identified paths are the primary sources for decision signals, as evidenced by a marked decrease in self-repair compared to non-path components; and (ii) transformers utilize modular, class-shared path mechanisms to process categorical information. Beyond these findings, we show that the potential implications of our method can stem from sparse manipulation, highlighting efficient applicability to downstream tasks such as model debugging and pruning. Code is available at `https://github.com/Jo-won/CausalPathTracing`.

1 Introduction

With the success of transformers [1] across language and vision, interest has grown in understanding their internal mechanisms beyond their black-box nature, especially to enable safer deployment in high-stakes applications such as healthcare, law, and education. Mechanistic interpretability aims to identify which internals of a model—such as attention heads or MLPs—contribute to its decisions.

In particular, building on circuit discovery in toy settings [2], a line of work has leveraged Pearl's causal theory [3] to explore internal causal structures. Depending on the granularity of ablation-based interventions, prior work can be classified into: *node-level patching* [4–9], which identifies the role of individual internals; *edge-level patching* [10–13], which examines the influence from one internal to another; and *path-level patching* [14–16], which traces internals connecting the input to the decision.

Recent work [17] has shown that ablation-based methods often fail to estimate causal effects reliably enough to reproduce a decision, due to *self-repair* (or backup behavior; see Appendix Section H.2), where later components compensate for earlier ablations. This implies that whenever unablated components lie between the target and the decision, internal explanations may be misattributed. To address this, one solution is to trace the full chain of internals connecting the input to the decision—what we term *causal referencing*.

However, prior work across all three categories above suffers from the following limitations with respect to causal referencing. At the definitional level, although these methods aim to explain instance-level causes for a single decision in a classification setting, they either lack an explicit instance-level causal definition or rely solely on counterfactual tests, making their explanations insufficient to

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Approach	Patching	Path Tracing for Decision	
		Feasibility	**Reliability**
[4–9]	Node	✓ backward chaining only	✗ no causal referencing
[10–13]	Edge	✓ backward chaining only	✗ no causal referencing
[14]	Path	✗ hypothetical test only	✓ full coverage
[15, 16]	Path	✗ manual subpath	✓ full coverage
CPT (Ours)	Path	✓ polynomial behavior on average	✓ full coverage

Table 1: **Comparison of patching methods for path tracing.** Feasibility: tractable computational cost in practice. Reliability: sufficiency to reproduce the decision. Backward chaining: concatenation of tracing results obtained independently at each block without causal referencing (Definition 3).

reproduce the decision. At the methodological level, node- and edge-level methods typically test only one or two internals independently, without causal referencing. As a result, they fail to capture set-level interactions among internals, where the joint effect of a set can differ from the sum of individual effects. Although concatenating such independent tests may yield a feasible explanatory chain from input to decision, the resulting chain is unreliable for reproducing the decision itself. Path-level methods can, in principle, support causal referencing while accounting for set-level interactions. However, their combinatorial complexity limits existing approaches to hypothetical interventions [14] or manual analysis of a single subpath [15, 16]. This makes it infeasible to capture a full explanation for a decision (Table 1).

In this work, we propose an automated path tracing method with causal referencing for reliable (i.e., sufficient to reproduce the decision) explanations within feasible runtime. We first unfold the internal computations of each block into additive path nodes. Then, we introduce an instance-level causal definition tailored to transformer internals, building on the Halpern–Pearl (HP) framework [18, 19]. Beyond the necessity captured by counterfactual tests, our definition further characterizes whether a set of internals is sufficient to reproduce the decision and, when jointly evaluated with these two conditions, minimal to rule out trivial solutions; the formulation is lifted to the *node-set level* to capture joint interactions rather than single-node effects. Building on this definition, we present two efficient strategies that operate on it: a minimality-based subset search for identifying causal node sets within each block, and a union-based referencing strategy that extends tracing across the entire model. Together, these two strategies, unlike existing approaches, enable a reliable path-level analysis of the internals responsible for a given decision, while running in polynomial time in practice.

Through extensive experiments, we empirically validate our approach and uncover the following findings about the traced paths. First, we observe that the traced paths are less subject to self-repair and instead act as self-repair givers, indicating that the primary sources of decision signals are concentrated along these paths. Second, we find that traced paths capture internal mechanisms shared across class-wise decisions—here, across image classes. Beyond these findings, we further demonstrate the practical implications of our approach, suggesting that it can serve as a foundation for future work in directions such as model debugging and pruning.

2 Methodology

2.1 Problem Setup in Transformer

We begin by introducing the problem setup in a standard transformer. Our goal is to trace the internal paths responsible for a given decision; as in prior work summarized in Table 1, we restrict our analysis to a single decision in a classification setting, and formally define it as follows:

Definition 1 (Model Decision)**.** *Let $y \in \mathbb{R}^{\mathcal{C}}$ be the model's output over $\mathcal{C}$ classes. We define the* ***model decision*** *as the index c^* such that $y^{(c^*)} > y^{(i)}$ for all $i \neq c^*$, a strict argmax.*

To produce such a decision, a transformer is composed of a sequence of blocks with an identical architecture. We denote each block by its input (z_{ib}) and output (z_{ob}) as follows:

$$\begin{aligned}
&\left[[z_q^{(h)}]_{h=1}^{H};\ [z_k^{(h)}]_{h=1}^{H};\ [z_v^{(h)}]_{h=1}^{H} \right] = \mathtt{L}_{ia}(\mathtt{LN}_1(z_{ib})),\\
&z_{oa} = \mathtt{L}_{oa}([\,\mathrm{softmax}(z_q^{(h)} z_k^{(h)\top}/\sqrt{d_h}) z_v^{(h)}\,]_{h=1}^{H}), \quad z_{im} = z_{ib} + z_{oa},\\
&z_{om} = \mathtt{L}_{om}\left(\phi(\mathtt{L}_{im}(\mathtt{LN}_2(z_{im})))\right), \quad z_{ob} = z_{im} + z_{om},
\end{aligned} \tag{1}$$

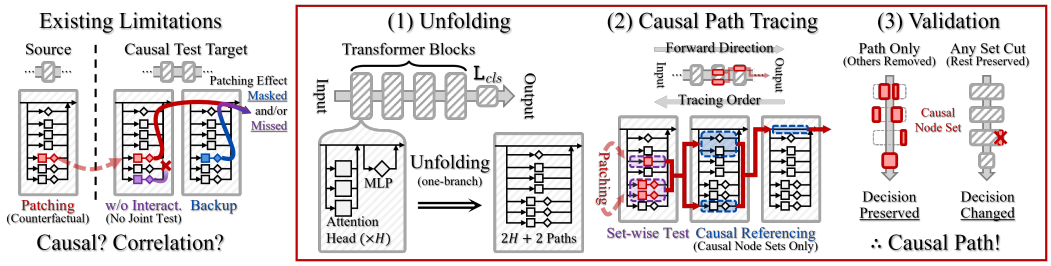


Figure 1: **Overview of our Causal Path Tracing (CPT).** Existing methods often fail to reliably capture causal effects as they do not account for interactions and self-repair. Our method unfolds transformer blocks and traces causal paths by considering interactions through set-wise testing and mitigating self-repair through causal referencing.

where $\texttt{LN}_*$ is layer normalization, $\texttt{L}_*(z) = zW_*^\top + b_*$ is linear projection, and ϕ is GeLU. Intermediate representations are $z_{ib}, z_{oa}, z_{im}, z_{om}, z_{ob} \in \mathbb{R}^{T\times d_m}$ and $z_q^{(h)}, z_k^{(h)}, z_v^{(h)} \in \mathbb{R}^{T\times d_h}$, where T is the number of tokens, d_m the model dimension, and $d_h = {}^{d_m}/_{H}$. Accordingly, the final block output is passed to the classifier $\texttt{L}_{cls}$ to produce decision y.

2.2 Unfolding Transformer Block

Within our framework, the granularity at which internal representations are viewed as nodes determines the granularity of tracing. A naive choice is to treat every intermediate representation z_* as an individual node, which exposes all internal computations—including specific attention heads, MLP, and residual paths—as candidate causes of the model's decision. However, this yields a highly branched graph and makes path tracing laborious.

To avoid this, we unfold each transformer block into a one-branch structure: given a block input z_{ib}, we decompose the block output z_{ob} into additive path terms with respect to z_{ib}, and treat each term as a single node. The decomposition groups all intermediate computations according to (i) which attention head, if any, they pass through, and (ii) whether they pass through the MLP. Obtaining this grouping requires expanding Equation (1) with respect to z_{ib}, which is nontrivial due to the non-linearities softmax and GeLU ϕ.

We resolve this by reinterpreting each non-linearity as a *soft gating transformation*. From an interpretative perspective, if the input and output of a non-linear function are retained in memory during a single forward pass, the input–output relation can be expressed as a Hadamard product with a matrix obtained by element-wise division of output by input. We call this matrix a *soft gating transformation matrix*. Softmax and GeLU are then reexpressed as $\mathrm{softmax}(z/\sqrt{d_h}) = z \odot D_\alpha$ and $\phi(z) = z \odot D_\beta$, and layer normalization can also be expressed through a memory-based reinterpretation as $\texttt{LN}(z) = zW_{ln}^\top + b_{ln}$. This does not linearize the block; it provides a linear interpretation of the block's computation *conditioned on cached representations from a forward pass*.

$$
\begin{aligned}
z_{oa} &= \Big(\sum_{h=1}^{H} z_q^{(h)} z_k^{(h)\top} \odot D_\alpha z_v^{(h)} W_{oa}^\top\Big) + b_{oa},\\
z_{om} &= (z_{ib} + z_{oa})\, W_{ln_2}^\top W_{im}^\top \odot D_\beta W_{om}^\top + b_{ln_2} W_{im}^\top \odot D_\beta W_{om}^\top + b_{im} \odot D_\beta W_{om}^\top + b_{om},\\
z_{ob} &= z_{ib} + z_{oa} + z_{om}\\
&= \underbrace{z_{ib}}_{\text{Residual Only (1 Path)}} + \underbrace{\sum_{h=1}^{H} z_q^{(h)} z_k^{(h)\top} \odot D_\alpha z_v^{(h)} W_{oa}^\top + b_{\text{attn}}}_{\text{Attention Only (H Paths)}} + \underbrace{z_{ib} W_{ln_2}^\top W_{im}^\top \odot D_\beta W_{om}^\top + b_{\text{mlp}}}_{\text{MLP Only (1 Path)}}\\
&\quad + \underbrace{\sum_{h=1}^{H} z_q^{(h)} z_k^{(h)\top} \odot D_\alpha z_v^{(h)} W_{oa}^\top W_{ln_2}^\top W_{im}^\top \odot D_\beta W_{om}^\top + \frac{b_{oa} W_{ln_2}^\top W_{im}^\top \odot D_\beta W_{om}^\top}{H} + b_{\text{attn+mlp}}}_{\text{Attention+MLP (H Paths)}},
\end{aligned}
\tag{2}
$$

Under this reinterpretation, z_{ob} admits an additive decomposition with respect to z_{ib} (Equation (2)), and its terms partition into $2H + 2$ groups indexed by which layer parameters they involve—attention

head, MLP, both, or neither. We take these *unfolded path terms* as the nodes in a block. The parameter-only terms b_{attn}, b_{mlp}, and $b_{\text{attn+mlp}}$ are assigned to their corresponding layers; the full derivation, including the unfolding of $z_q^{(h)}$, $z_k^{(h)}$, $z_v^{(h)}$ (which share the path structure of z_{ib} via W_{ln_1} and W_{ia}), is given in Appendix Section F. Note that, in implementation, D_α is handled by passing through the forward pass without issue; in the case of D_β, since $\phi(z)/z \to 0.5$ as $z \to 0$, it uses a threshold-based fallback to avoid division by zero.

2.3 Causal Definitions for Transformer Internals

We view a transformer as a computational graph in which each node represents an internal activation. In our formulation, each unfolded term in Equation (2) corresponds to a path node, named according to the modules involved in producing its activation. For example, an MLP-only node represents the path from the block input to its output that passes through the MLP but not attention (Figure 1).

A node set consists of child path nodes sharing a common parent within a single block, capturing interactions among nodes that cannot reach one another through directed edges. We denote the collection of all such node sets by $\mathcal{V}$ and the collection of all paths, including subpaths, by $\mathcal{P}$.

Definition 2 (Path)**.** *We define a **path** as a set of node sets with exactly one node set per block, spanning all blocks. A **subpath** is one that spans a contiguous subset of blocks.*

Definition 3 (Subpath Reference and Referencing)**.** *Given a node set V, we define **a subpath reference** as a downstream subpath connecting V to the output, each serving as a reference for evaluating the effect of V. Thus, **referencing** for V means considering its subpath references.*

The effect of V on the decision is evaluated through a downstream subpath rather than in isolation. Since multiple downstream subpaths may connect V to the output, the evaluated effect can differ depending on the chosen reference. With this notion in place, we now define what it means for a node set to be *causal* for a decision, evaluated relative to a subpath reference.

Definition 4 (Causal Node Set)**.** *For a node set $V \in \mathcal{V}$ in a transformer, let $P \in \mathcal{P}$ be a subpath reference connecting V to the output, and an off-path node set $\hat{V}$ such that $P \cup \hat{V}$ equals the set of all nodes except V and its ancestors, and $P \cap \hat{V} = \emptyset$. We say V is **causal** for the decision with respect to such P if it satisfies*

*(a) **Necessity (Counterfactual)**: Let V' denote that V is intervened on to take a different value, and let $\hat{V}'$ be defined analogously for $\hat{V}$ to causally isolate V. Under these interventions, the output y' satisfies $\arg\max_i y'^{(i)} \neq c^*$, where c^* denotes the decision without any intervention.*

*(b) **Sufficiency (Contingency)**: Given V, even if nodes in P are perturbed due to an intervention on $\hat{V}'$, the output y'' satisfies $\arg\max_i y''^{(i)} = c^*$.*

where P inductively satisfies:

*i. **Base Case**: If the only path from V to the output passes through classifier L_{cls} alone, then P reduces to $\{L_{cls}\}$*

*ii. **Inductive Case**: Otherwise, P is a subpath reference whose constituent node sets have already been identified as causal starting from the base case, connecting V to the output. We call such P a **causal subpath reference.***

Necessity ensures that if V is causal, removing it must alter the decision—a node set that is dispensable cannot be considered a cause; *Sufficiency* ensures that the decision is preserved even when perturbations are induced along the downstream causal subpath by intervening on sibling nodes of nodes in V, reflecting the robustness of V as a causal contributor.

Remark 1 (Minimality of Causal Node Set)**.** *A causal node set V is said to satisfy **minimality** if no strict subset of V satisfies both conditions (a) and (b). Note that a node set failing to satisfy minimality is not thereby rendered non-causal. However, when evaluating whether a target node set is causal, minimality should be considered jointly with both conditions (a) and (b); otherwise, the entire model trivially satisfies both conditions.*

The inductive construction of P starts from the classifier L_{cls}, whose use as a reference requires no prior causal definition. This eliminates circularity. Without causal referencing, one cannot guarantee that an identified node set is reliably causal for the model's decision, since later components (i.e., nodes in $\hat{V}$) may compensate for the intervened effect (see Appendix Example 1). For causal evaluation of the target node set V, one node set is selected among combinations of path nodes sharing a common parent presented in the preceding section; the remaining sibling nodes with respect to the parent of V are then intervened on to isolate the causal effect of V alone.

Algorithm 1 Minimality-based Causal Subset Search per Block

Input: The set of all path nodes within a block $V_p = [v_1, \ldots, v_n]$, subpath reference P connecting V_p to the decision c^* in the output y, and an off-path node set $\hat{V}$ such that $P \cup \hat{V}$ equals the set of all nodes except the ancestors of V, and $P \cap \hat{V} = \emptyset$
Output: $V_{\text{out}} = \{V_1, V_2, \ldots\}$, where each $V_i \subseteq V_p$ satisfies Conditions in Definition 4
$V_{\text{out}} \leftarrow \emptyset$
for $s = 1$ to n **do** ▷ Subset size
  for each $V \subseteq V_p$ such that $|V| = s$ **do**
    if $V_i \subseteq V$ for some $V_i \in V_{\text{out}}$ **then**
      continue ▷ Fail Remark 1
    end if
    Intervene on $V \to V'$, and on $\hat{V} \to \hat{V}'$
    Let $y' \leftarrow$ model output under $(V', \hat{V}', P)$ ▷ Condition (a) in Definition 4
    Let $y'' \leftarrow$ model output under $(V, \hat{V}', P)$ ▷ Condition (b) in Definition 4
    if $\text{argmax}_i\, y'^{(i)} = c^* \vee \text{argmax}_i\, y''^{(i)} \neq c^*$ **then**
      continue ▷ Fail (a) or (b)
    end if
    $V_{\text{out}} \leftarrow V_{\text{out}} \cup \{V\}$ ▷ Satisfies (a) and (b) with Remark 1
  end for
end for
return V_{out}

Operationally, we implement the intervention on V and $\hat{V}$ by replacing its value with that of the corresponding position from tokens that produce a different decision (as in Pearl's do-operator). This grounds the intervention in the model's actual representational space rather than arbitrary perturbation. Building on these definitions, we now introduce the central object of our analysis by chaining causal node sets across transformer blocks:

Definition 5 (Causal Path)**.** *We define a **causal path** as a path in which each node set is causal.*

2.4 Minimality-based Causal Subset Search per Block

From Section 2.2, each block contributes $2H + 2$ additive path nodes; we denote the set of all path nodes within a block by V_p. Definition 4 then requires identifying every $V \subseteq V_p$ satisfying necessity and sufficiency. Subset-level evaluation is required to consider joint interaction effects among nodes on the decision, where the effect of a node set is not the sum of its individual node effects. Naively, this incurs $2^{|V_p|}$ candidates, which is precisely why prior path-level methods have been infeasible.

Our key observation is that the trivial-satisfaction phenomenon of Remark 1 doubles as a pruning rule: once V is confirmed causal, every strict superset of V violates minimality and can be skipped. We thus search bottom-up by subset size $s = 1, 2, \ldots, n$, where $n = |V_p| = 2H + 2$ (Algorithm 1), pruning supersets only *after* a smaller causal set has been confirmed, so no minimal set is preemptively discarded. The worst case remains $2^{|V_p|}$, but Theorem 1 shows that under a mild probabilistic abstraction over whether a subset is validated as causal, the expected number of evaluations is polynomial (proof in Appendix).

Theorem 1 (Expected Number of Subset Evaluations under Minimality Search)**.** *Consider a minimality-based subset search over n nodes, where each subset is independently selected as a causal node set with probability p under an abstract probabilistic model. Then, the expected number of subset evaluations over all subsets is estimated by:*

$$n + (1-p)\sum_{s=2}^{n} \max\left(0, \binom{n}{s} + \sum_{i=1}^{s-1} \sum_{m=1}^{\lfloor p\binom{n}{i} \rfloor} (-1)^m \binom{\lfloor p\binom{n}{i} \rfloor}{m} \binom{n-mi}{s-mi}\right),$$

Under this, the expected number of evaluated subsets is upper-bounded approximately by:

$$O\left(n^{\lfloor \log_2(\frac{1}{p}+2) \rfloor}\right). \tag{3}$$

Remark 2. *The exact value of p is generally unknown and task-dependent, and we do not assume it to be small. Rather, the analysis highlights that the conditions required for significant pruning—and hence reduced average search cost—are mild, excluding only highly degenerate regimes such as*

Algorithm 2 Unfolded Block-wise Causal Path Tracing

Input: A transformer with D blocks; model output y with decision c^* for a given input
Output: A causal path P, obtained as the union of causal paths across all blocks
$P \leftarrow (\{\mathtt{L}_{cls}\})$ ▷ Initial subpath reference
for $j = D, D-1, \ldots, 1$ **do** ▷ Iterate backward
  $V_p^{(j)} \leftarrow$ unfolded path nodes in block j
  $V_{\text{out}}^{(j)} \leftarrow \textsc{Min_Search}(V_p^{(j)}, P, c^*)$ ▷ Algorithm 1
  $P \leftarrow P \cup \left(\bigcup V_{\text{out}}^{(j)}\right)$ ▷ Extend subpath reference toward the input
end for
return P

$p \leq \frac{1}{2^n-2}$. *Empirically, we observe that these conditions are frequently satisfied in practical models, resulting in polynomial-time behavior on average. Worst-case exponential complexity remains unavoidable in principle due to the* 2^n *search space.*

2.5 Unfolded Block-wise Causal Path Tracing

Building on the per-block search of Section 2.4, we extend tracing to the full transformer. Since tracing must proceed in the order of connection to the decision, we traverse blocks backward, $j = D, \ldots, 1$, starting from the classifier $\mathtt{L}_{cls}$ and growing the reference one block at a time toward the input.

A direct backward extension is, however, infeasible. As a consequence of considering node-set-level, a single block may admit *multiple* causal node sets, each spawning distinct causal subpath references that are independently necessary and sufficient for the decision. Reliable tracing then requires running the per-block search once per reference at the next block, and the number of references multiplies across blocks—yielding a branching tree whose size grows exponentially in D.

Notably, the union of all causal node sets across these references functions as a single causal reference. By ensuring this union satisfies the causal criterion directly, we reduce the complexity from exponential to linear (Algorithm 2). Theorem 2 formalizes this: under a mild probabilistic model, the reliability of union-based referencing—defined as the probability that identified node sets preserve the decision—converges to 1 (proof in Appendix).

Theorem 2 (Causal Union Reference Reliability)**.** *Consider a minimality-based subset search over* n *nodes, where each subset is independently selected as a causal node set with probability* p *under an abstract probabilistic model. Suppose that a collection of such sets,* $V_{out}^{(j+1)} = \{V_i^{(j+1)}\}_{i=1}^k$*, is identified as causal in the* $(j+1)$*-th block. Their union, denoted as* $P = \bigcup_{i=1}^k V_i^{(j+1)}$*, serves as the causal subpath reference for the minimality-based subset search in the* j*-th block. Let* s_{avg} *denote the average size of the* k *causal node sets in* $V_{out}^{(j+1)}$*. Then, the reliability of the resulting causal node set obtained using* P *is given by:*

$$p + (1-p)\left(1 - \left(1 - \frac{s_{avg}}{n}\right)^k\right)^n \to 1 \tag{4}$$

3 Experiments

Models and Datasets. We use three language models (GPT2-xs [20], Pythia-14m, Pythia-1b [21]) and two vision models (ViT-tiny [22], DeiT-tiny [23]). For language tasks, we use KNOWNS1000 [6] and T-REX [24, 25] datasets; for vision tasks, IMAGENET [26] and OFFICEHOME [27].

Baselines. We compare with existing methods as summarized in Table 1. To enable fair comparison with our method, all baselines are extended under a backward chaining framework, assuming that residual connections are always present. Specifically, NT_1 and $\text{NT}_{10\%}$ are adaptations of the node-level method [6], selecting the top-1 node per block or the top 10% of nodes across all blocks as decision paths. ET_{all} and ET_{cls} are adaptations of edge-level method [12], selecting decision paths based on task-level attribution; the former defines the task over the entire dataset, while the latter does so per class. $\text{ET-IG}_{\text{all}}$ and $\text{ET-IG}_{\text{cls}}$ further extend these with integrated gradients [28], based on the edge-level method [13]. Note that since existing path-level methods are computationally infeasible for exhaustive tracing, we separately compare CPT with early-terminated adaptations of them in Appendix Section H.11. We refer to our method as Causal Path Tracing (CPT).

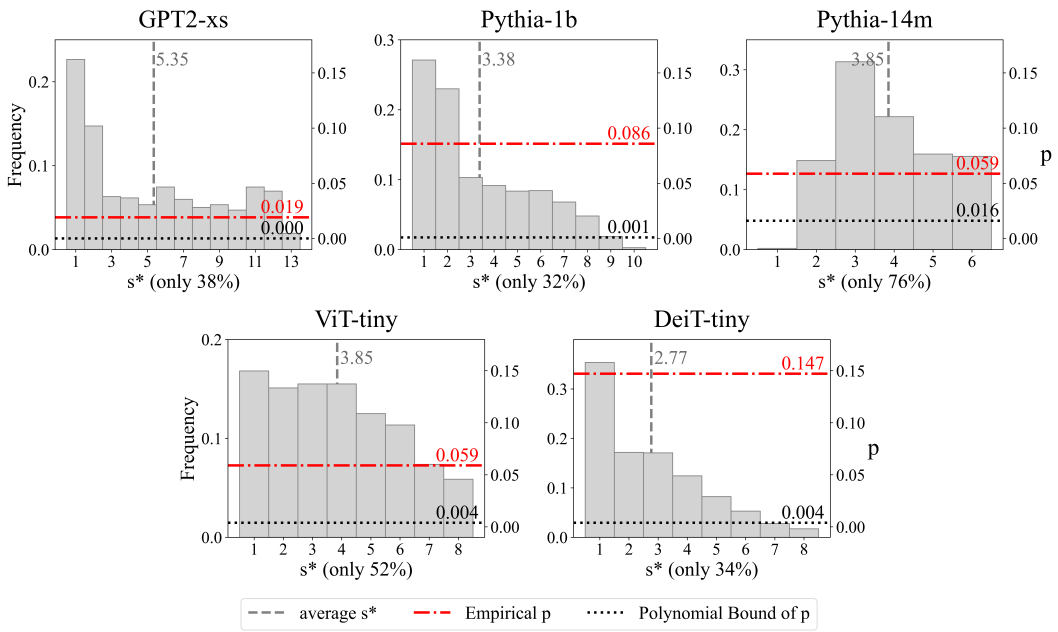


Figure 2: **Comparison of empirical time complexity.** Empirical polynomial-time behavior across language models on T-REx and vision models on IMAGENET. Subplots denote reduced search spaces (parentheses); s^* indicates the maximum minimality-based search step (Equation (3)); p represents the empirical estimate from average s^* (Theorem 1) alongside its theoretical polynomial bound (Remark 2).

3.1 Empirical Validation of the Proposed Method

How Does Each Component Affect CPT? We conduct an ablation study on Pythia-14m using KNOWNS1000 to quantify the contribution of each CPT component. As shown in Table 2, union-based referencing merges multiple downstream subpath references into a single reference, reducing referencing trials and yielding about 2.7× speedup. Minimality-based search synergistically reduces both within-block subset evaluations and downstream referencing trials by eliminating redundant causal node sets, providing about 1.14× speedup on top of union-based referencing and about 27.3× speedup when both are combined. Unfolding is retained in all configurations, as it transforms multi-branch transformer blocks into a one-branch structure that enables both search strategies. Overall, these components play complementary roles in substantially reducing computational cost while maintaining high hit rates.

Method	Min.	Union	Runtime(s)	Hit rate
CPT	✓	✓	**10.04**	0.970
w/o Union	✓	✗	26.97	1.000
w/o Min.	✗	✓	11.45	1.000
w/o Both	✗	✗	273.83	1.000

Table 2: **Ablation study of CPT components.** Min.: minimality-based search; Union: union-based referencing. Runtime and hit rate on Pythia-14m with KNOWNS1000.

Does Our Method Exhibit Polynomial Behavior Empirically? Figure 2 presents the empirical time complexity of our causal path tracing procedure across models. Each subplot shows the distribution of s^*, defined as the final step in the minimality-based search where no further superset remains due to pruning by already selected causal node sets. The average s^* is used to estimate the empirical probability p under the probabilistic abstraction in Theorem 1, which is then compared against the lower bound suggested by the analysis for polynomial-time behavior (see Remark 2).

Across all models, the empirical p exceeds the analytical threshold, indicating substantial pruning in the evaluated settings. Pruning often concludes well before the search space midpoint, with the search typically completing in few steps. The distribution of s^* further elucidates the model's internal decision-making: a small s^* concentrated near one indicates reliance on individual path strengths, whereas a larger or more dispersed s^* suggests that reasoning emerges from interactions among multiple paths rather than any single dominant one. To complement this complexity analysis, we

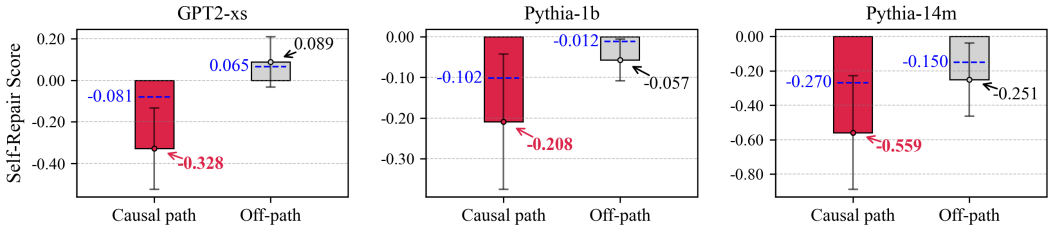


Figure 3: **Self-repair scores on causal path vs. off-path.** Each bar shows the mean (dot with arrow) and standard deviation (error bar); medians are shown as blue dashed lines. Lower scores indicate less self-repair. Results are averaged over KNOWNS1000 and T-REX.

further examine actual wall-clock runtime across model sizes (Appendix Figure D). We also evaluate CPT on models up to Pythia-12b under a restricted search budget, reporting runtime and GPU memory usage (Appendix Section H.8).

Does the Traced Path Preserve the Decision More Reliably than Baselines? Each value in Table 3 represents the average score across models on two datasets. All methods are evaluated by pruning the model to retain only the paths identified by each method. We report three metrics: (i) **Hit.** (hit rate) measures the proportion of cases in which the pruned model produces the same decision as the original. A value close to 1 indicates that the traced path reliably captures the model's decision. (ii) $\mathbf{PS}_t$ (path node sparsity) denotes the fraction of path nodes in the traced path relative to all path nodes in the model. (iii) $\mathbf{FS}_t$ (FLOPs sparsity) denotes the ratio of FLOPs consumed by the traced path to the total FLOPs of the model. Both metrics measure retained fractions; lower values therefore indicate sparser paths.

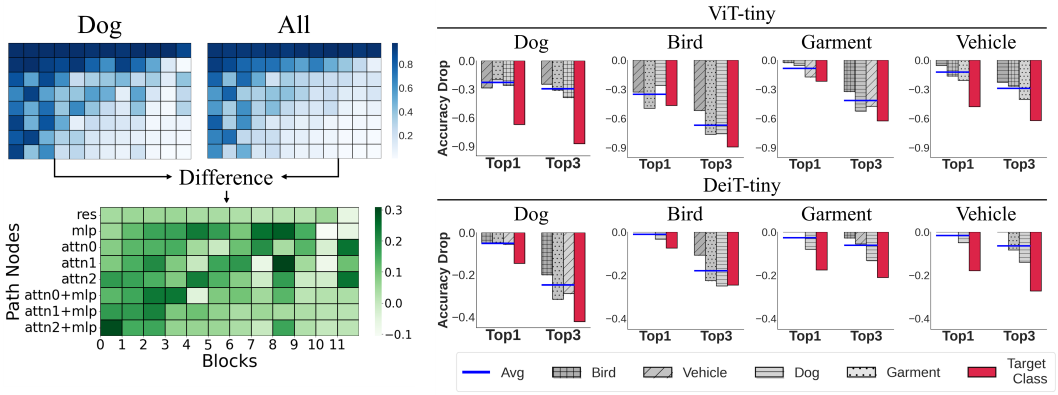
	Language			Vision		
Method	**Hit.** (↑)	$\mathbf{FS}_t$	$\mathbf{PS}_t$	**Hit.** (↑)	$\mathbf{FS}_t$	$\mathbf{PS}_t$
NT_1	0.000	0.018	0.113	0.008	0.004	0.125
$NT_{10\%}$	0.000	0.042	0.121	0.009	0.000	0.125
ET_{all}	0.211	0.936	0.758	0.426	0.977	0.784
ET_{cls}	0.416	0.963	0.762	0.261	0.936	0.717
ET-IG_{all}	0.689	0.987	0.859	0.382	0.974	0.763
ET-IG_{cls}	0.634	0.974	0.806	0.287	0.929	0.713
CPT	**0.981**	0.778	0.578	**0.964**	0.693	0.535

Table 3: **Quantitative validation of decision preservation.** Average performance. **Hit.** (hit rate) measures whether the pruned model preserves the original decision; $\mathbf{FS}_t$ denotes the fraction of computation retained by the identified path; $\mathbf{PS}_t$ denotes the fraction of path nodes retained.

Our method (CPT) achieves a near-perfect hit rate, consistent with the analysis in Theorem 2 that the identified paths are reliably causal. In contrast, existing methods show substantially lower hit rates, supporting our claim in Table 1 that while tracing is feasible with backward chaining, it is generally not reliable for identifying decision-relevant paths. In particular, the NT-based methods retain very small fractions of FLOPs and path nodes, with hit rates close to zero, whereas ET and ET-IG attain hit rates that remain low despite their traced paths consuming nearly the total FLOPs of the model and covering a large number of path nodes. These results indicate that NT-based methods incur many false negatives in identifying internals that are genuinely causal to the decision, while ET and ET-IG suffer from many false positives. Unlike these baselines, our method achieves both lower false positives than ET and ET-IG and lower false negatives than NT, demonstrating that CPT is the most reliable choice among the compared approaches. We further compare CPT with gradient-based attribution methods, where CPT also achieves substantially higher decision preservation (Appendix Section H.10).

3.2 Experimental Findings and Mechanistic Insights

Traced Paths are Less Self-Repaired and Instead Act as Self-Repair Givers We compare self-repair scores between attention heads on the causal path and those off the path, as identified by our tracing method. Following [17], we categorize components based on whether they belong to the traced causal path and measure their self-repair accordingly. Here, the self-repair score measures how much a node acts as a backup receiver, i.e., how strongly it is compensated by other nodes during ablation, rather than acting as a compensating (giver) node for others (Appendix Sections H.2 to H.4).



(a) (b)

Figure 4: **Causal paths uniquely activated for specific classes.** *(a)* Average causal path ratios for target (left) and all classes (right), with their difference (bottom) highlighting class-specific paths. res, mlp, attn#: residual, MLP, and attention head # paths. *(b)* Accuracy drop upon ablating the most class-specific path, showing selective reliance (see Appendix Section H.5).

As shown in Figure 3, self-repair occurs less frequently on the traced causal path. Despite the inherent noise in self-repair [17], both mean and median scores are consistently lower on-path than off-path. This suggests that fewer on-path nodes act as receivers; given the fixed total logit, these nodes function primarily as information sources (givers) for the decision. Consequently, the traced path contains a higher concentration of necessary and sufficient causal sources compared to nodes outside the path.

The Traced Path Captures Internal Mechanisms Shared Across Class-Wise Decisions Furthermore, we aim to investigate whether the discovered causal paths capture shared mechanisms across class-wise decisions. We select four super-classes—dog, bird, garment, and vehicle—based on WordNet semantic similarity and aggregate causal paths from individual samples to analyze subpath frequencies. Compared to the global average, certain subpaths are consistently selected and exhibit significantly higher activity within specific super-classes (Figure 4(a)). These are referred to as class-wise causal subpaths, and it is hypothesized that *they store key discriminative information relevant to their super-classes* due to their unusually high activation rates.

To validate this hypothesis, we intervene on the class-wise causal subpaths and measure the resulting performance drop, expecting that removing class-specific information leads to a larger accuracy degradation within the corresponding super-class. As shown in Figure 4(b), this pattern is clearly observed: for example, deactivating the dog class-wise causal subpaths in a ViT-tiny model reduces the top-1 accuracy for dog samples by approximately 44.7% more than for other super-classes, with similar trends across bird, garment, and vehicle classes. While semantic overlap among IMAGENET classes and visual diversity within each super-class prevent complete performance collapse, the consistent and pronounced class-wise degradation across all super-classes indicates that our method effectively identifies causal subpaths that play a meaningful role in class-specific inference.

3.3 Potential Implications for Practical Applications

How Sparse Can a Path Intervention Be While Flipping a Decision? Computations inside a transformer proceed layer by layer in a forward direction without cycles, naturally forming a directed acyclic graph in which each node depends only on its parents and is conditionally independent of its non-descendants once the parent values are fixed. By virtue of this property, if a path explaining a decision has been correctly identified, then intervening on all of its nodes residing in a single block should suffice to flip that decision. To verify this, we select, for each method's traced path, the block containing the fewest path nodes, and intervene only on its nodes in that block—leaving the rest of the model untouched. We then observe whether the model's decision flips.

As shown in Table 4, our method flips the original decision almost perfectly (hit rate near 0) with highly sparse interventions—only 3–5% in $\mathbf{FS}_t$ and 1–3% in $\mathbf{PS}_t$. Compared to ET- and ET-IG-based methods, which achieve comparable decision flipping while intervening on roughly twice the fractions of FLOPs and path nodes, our method requires smaller interventions. Compared to NT-based methods,

	Language			Vision		
Method	**Hit. (↓)**	FS_t	PS_t	**Hit. (↓)**	FS_t	PS_t
NT_1	0.617	0.006	0.015	0.187	0.027	0.010
$NT_{10\%}$	0.569	0.002	0.015	0.187	0.027	0.010
ET_{all}	0.116	0.103	0.059	0.008	0.102	0.042
ET_{cls}	0.020	0.123	0.069	0.023	0.085	0.035
ET-IG_{all}	0.000	0.131	0.077	0.020	0.102	0.042
ET-IG_{cls}	0.000	0.114	0.068	0.153	0.081	0.035
CPT	**0.000**	**0.050**	**0.031**	**0.011**	**0.034**	**0.012**

Table 4: Quantitative comparison of decision-flipping efficiency via sparse path manipulations. Ours in bold.

Model	**FS_t**	**PS_t**
GPT2-xs	0.659 (-7.1%)	0.406 (-28.5%)
Pythia-1b	0.650 (-3.4%)	0.362 (-20.6%)
Pythia-14m	0.950 (-0.0%)	0.678 (-4.6%)
Avg.	0.753 (-3.2%)	0.482 (-16.6%)
ViT-tiny	0.619 (-18.5%)	0.441 (-23.3%)
DeiT-tiny	0.566 (-16.8%)	0.389 (-21.3%)
Avg.	0.593 (-17.7%)	0.415 (-22.4%)

Table 5: Potential for further sparsification of CPT paths while preserving decisions.

which use similarly sparse interventions, our method flips decisions far more reliably. Together, these results indicate that the paths identified by our method serve not only as explanatory traces but also as practical substrates for downstream applications—providing a foundation for debugging or correcting systematic failure modes, though directing the flipped decision toward a target remains future work.

How Sparse Can a Traced Path Be While Still Preserving a Decision? Sparsity and reliability exist in tension: our union-based referencing prioritizes reliability, preserving decisions at the cost of compactness. We thus view further sparsification as a complementary direction rather than a limitation. However, this trade-off does not imply an optimal sparsity level: the union operator, used to avoid the exponential cost of enumerating minimal subsets, may retain overlapping causal contributions, leaving room for additional compression.

To indicate a concrete direction for future work, we heuristically examine potential sparsity gains while preserving the decision. Specifically, for identified causal node sets, we enumerate subsets by size and select the smallest subset that maintains the decision, shrinking the per-block union. As shown in Table 5, this reduces the retained-FLOPs fraction (**FS_t**) by up to 18.5% and the retained-node fraction (**PS_t**) by up to 28.5%, while preserving the hit rate. These results suggest that a principled, non-heuristic refinement could extend our framework from reliable causal attribution to efficient instance-level pruning.

4 Conclusion

In this paper, we presented an automated path tracing method for a decision. By enabling reliable and feasible tracing of decision-relevant causal paths in practice, our method moves mechanistic interpretability from localized inspection toward path-level causal explanations. This opens up new directions for developing causal tools for model debugging and safety, as well as for task-specific (i.e., class-specific) model pruning informed by sparse path manipulation and shared internal path-level mechanisms. As future work, we aim to move beyond the limitations discussed in Appendix Section B by extending our method to larger models, multi-token output settings, and the analysis of how causal paths emerge during training. Through these directions, we believe that this line of research can make transformers more transparent and safer for deployment in safety-critical domains, while also enabling more efficient adaptation to target tasks with reduced training time and computational cost.

Acknowledgments and Disclosure of Funding

This work was supported by the Institute for Information & Communications Technology Planning & Evaluation (IITP) grants funded by the Korea government (MSIT) (No. RS-2019-II190075, Artificial Intelligence Graduate School Support Program (KAIST); No. RS-2022-II220984, Development of Artificial Intelligence Technology for Personalized Plug-and-Play Explanation and Verification of Explanation; No. RS-2022-II220184, Development and Study of AI Technologies to Inexpensively Conform to Evolving Policy on Ethics; No. RS-2024-00457882, AI Research Hub Project; No. RS-2024-00509258, AI Guardians: Development of Robust, Controllable, and Unbiased Trustworthy AI Technology), and by the Korea Planning & Evaluation Institute of Industrial Technology (KEIT) grant funded by the Ministry of Trade, Industry and Energy (No. RS-2025-25458052, Development of Core Technologies for Manufacturing Foundation Models).

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

[2] Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. URL `https://transformer-circuits.pub/2021/framework/index.html`.

[3] Judea Pearl. *Causality: models, reasoning and inference*. Cambridge University Press, USA, 2 edition, 2009. ISBN 052189560X.

[4] Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. Investigating gender bias in language models using causal mediation analysis. *Advances in Neural Information Processing Systems*, 33:12388–12401, 2020.

[5] Kevin Ro Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the wild: a circuit for indirect object identification in GPT-2 small. In *International Conference on Learning Representations*, 2023.

[6] Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, 35:17359–17372, 2022.

[7] Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer. *The Eleventh International Conference on Learning Representations*, 2023.

[8] Stefan Heimersheim and Jett Janiak. A circuit for Python docstrings in a 4-layer attention-only transformer. *AI Alignment Forum*, 2023. URL `https://www.alignmentforum.org/posts/u6KXXmKFbXfWzoAXn/a-circuit-for-python-docstrings-in-a-4-layer-attention-only`.

[9] Fred Zhang and Neel Nanda. Towards best practices of activation patching in language models: metrics and methods. In *International Conference on Learning Representations*, 2024.

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

[11] Adithya Bhaskar, Alexander Wettig, Dan Friedman, and Danqi Chen. Finding transformer circuits with edge pruning. *Advances in Neural Information Processing Systems*, 37:18506–18534, 2024.

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

[13] Michael Hanna, Sandro Pezzelle, and Yonatan Belinkov. Have faith in faithfulness: Going beyond circuit overlap when finding model mechanisms. In *ICML 2024 Workshop on Mechanistic Interpretability*, 2024.

[14] Lawrence Chan, Adrià Garriga-Alonso, Nicholas Goldowsky-Dill, Ryan Greenblatt, Jenny Nitishinskaya, Ansh Radhakrishnan, Buck Shlegeris, and Nate Thomas. Causal scrubbing: a method for rigorously testing interpretability hypotheses. *AI Alignment Forum*, 2022. URL `https://www.alignmentforum.org/posts/JvZhhzycHu2Yd57RN/causal-scrubbing-a-method-for-rigorously-testing`.

[15] Michael Hanna, Ollie Liu, and Alexandre Variengien. How does GPT-2 compute greater-than?: Interpreting mathematical abilities in a pre-trained language model. *Advances in Neural Information Processing Systems*, 36:76033–76060, 2023.

[16] Neel Nanda, Senthooran Rajamanoharan, János Kramár, and Rohin Shah. Fact finding: attempting to reverse-engineer factual recall on the neuron level (Post 1). *AI Alignment Forum*, 2023. URL `https://www.alignmentforum.org/posts/iGuwZTHWb6DFY3sKB/fact-finding-attempting-to-reverse-engineer-factual-recall`.

[17] Cody Rushing and Neel Nanda. Explorations of self-repair in language models. In *Proceedings of the International Conference on Machine Learning*, pages 42836–42855, 2024.

[18] Joseph Y Halpern and Christopher Hitchcock. Actual causation and the art of modeling. *arXiv preprint arXiv:1106.2652*, 2011.

[19] Joseph Y Halpern. A modification of the Halpern–Pearl definition of causality. *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pages 3022–3028, 2015.

[20] Algorithmic Research Group. gpt2-xs, 2025. URL `https://huggingface.co/AlgorithmicResearchGroup/gpt2-xs`. Accessed: 2025-05-16.

[21] Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR, 2023.

[22] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[23] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021.

[24] Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. T-REx: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[25] Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, 2019.

[26] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115:211–252, 2015.

[27] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5018–5027, 2017.

[28] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, pages 3319–3328. PMLR, 2017.

[29] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019. URL `https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf`.

[30] Avanti Shrikumar, Peyton Greenside, Anna Shcherbina, and Anshul Kundaje. Not just a black box: Learning important features through propagating activation differences. *arXiv preprint arXiv:1605.01713*, 2016.

[31] Frauke Andersen, William Rudman, Ruochen Zhang, and Carsten Eickhoff. APP: Accelerated path patching with task-specific pruning. *arXiv preprint arXiv:2511.05442*, 2025.

A Related Work

Mechanistic interpretability aims to identify how internal components—such as attention heads and MLPs—contribute to model behavior. Early work explored circuit discovery in simplified settings [2], while recent methods adopt Pearl's causal theory [3] and ablation-based interventions to reveal components essential to specific outputs.

Node-level patching These methods aim to identify the contribution of individual input features to a model's prediction. [4] investigate gender bias through causal mediation analysis by selectively patching specific input features. [5] ablate features one by one to study their roles in indirect object identification, and further report the phenomenon of backup behavior, where other components compensate for ablated information. [6] restore specific token-level features to identify where factual knowledge is stored in the model when predicting objects in sentences, and demonstrate direct editing of such representations. Building on this, [7] extend the editing framework to perform large-scale knowledge editing across many memory locations. [8] analyze individual input features to uncover the mechanisms behind repeated argument names in Python docstring generation. [9] systematically examine the impact of various methodological choices in node-level patching, providing a comprehensive evaluation of this line of research.

Edge-level patching These methods examine the influence of neighboring feature pairs with direct computational dependencies. [10] propose a greedy edge ablation algorithm to automatically abstract model behavior by iteratively removing dependencies and observing the impact on predictions. [12] accelerate edge-level analysis by introducing a gradient-based first-order approximation that estimates the effect of ablating each edge efficiently. [13] further extend this with integrated gradients [28]. [11] replace missing edges with counterfactual activations obtained from corrupted examples, enabling gradient-based pruning without relying on discrete search or linear approximations.

Path-level patching These approaches investigate the contribution of distant feature pairs connected through multiple accumulated dependencies. [14] justify that ablations using zero or mean activations can induce misleading interventions, and instead propose hypothetical tests to explore decision paths. [15] manually specify and patch internal paths in a language model to explain the circuit responsible for a year-span prediction task. [16] manually ablate specific paths to investigate where factual knowledge related to sports is stored within the model.

When applying existing methods to trace paths for a given decision, node- and edge-level approaches are feasible under a backward chaining framework. However, they lack reliability as they perform chaining without causal referencing, making it difficult to guarantee the correctness of the identified paths. Path-level methods, on the other hand, often rely on hypothetical tests or require manual specification of paths, which renders full explanations infeasible due to the combinatorial complexity of possible paths. In contrast, our method enables reliable identification of causal paths while remaining computationally efficient, achieving tractable search on average in polynomial time.

B Limitations and Future Directions

Refining Linear Interpretations of Transformer Blocks. To enable efficient analysis under a one-branched structure, we employ a soft gating transformation to reinterpret non-linear functions with D_α and D_β. We note that D_β is involved in the MLP input representation, which itself aggregates both residual and attention outputs. As a result, the gating applied to the MLP-only path term may implicitly include influences that could otherwise be attributed to the attention+MLP path. However, this does not invalidate our path-level causal interpretation. The identified paths remain causally responsible for the model's decision, i.e., preserving or changing the decision. This is empirically supported by our quantitative results, which demonstrate *near-perfect hit rates* across tasks and models. Looking forward, a direction for future work is to move beyond interpretative linearization toward representations that are intrinsically closer to linear, enabling more refined disentanglement of nonlinear interactions while retaining the efficiency and causal reliability of path-level analysis.

Dependence on an Abstract Probability Model. Our analysis of the minimality-based subset search characterizes its expected behavior under a probabilistic abstraction in which each candidate subset is selected as causal with probability p. While this abstraction is not intended as a generative model of transformer internals, it serves to formalize how minimality-driven pruning can substantially reduce the effective search space. Importantly, the analysis does not claim worst-case polynomial-

time guarantees; exponential complexity remains unavoidable in principle. In practice, the value of p is task- and model-dependent and cannot be known beforehand. For this reason, we treat the theoretical analysis as providing sufficient conditions under which pruning is effective, rather than as a universal guarantee. Our empirical results consistently satisfy these conditions across models and tasks (Figure 2), suggesting that the abstraction captures a practically relevant operating regime of trained models. We further note that the i.i.d. assumption underlying this abstraction does not capture dependencies among candidate subsets: in transformer internals, overlapping subsets share components, and their causal status may therefore be correlated. Such dependencies may facilitate the early identification of minimal causal subsets, allowing larger portions of the search space to be pruned than the i.i.d. analysis predicts. In such cases, the i.i.d. analysis may provide a conservative estimate of the effective search cost. Nonetheless, developing tighter, model-aware characterizations of pruning behavior—potentially grounded in structural properties of trained transformers—remains an important direction for future work.

Decision-Level Causality versus Continuous Logit Attribution. Our framework adopts decision preservation and decision flip as the primary causal criteria, rather than relying on continuous changes in logits or probabilities. We make this choice to avoid introducing arbitrary thresholds into Boolean causal definitions. In our setting, notions such as sufficiency and necessity require unambiguous binary conditions, consistent with standard formulations of causality [3, 19]. If causality were instead defined through continuous logit variation, one would need to impose an arbitrary scalar threshold (e.g., determining how much logit decrease is "causally sufficient"). Conversely, treating any negative logit shift as causal would trivialize the explanation by attributing causality to a large fraction of dense transformer internals. In practice, this would again require thresholding, thereby reintroducing the same arbitrariness the formulation seeks to avoid. For this reason, we ground causality in the model's final decision, i.e., the argmax, which provides a clear and operationally well-defined criterion.

At the same time, we acknowledge that preserving a decision does not necessarily imply preserving the corresponding confidence or logit magnitude. For example, in our pruning-oriented analysis based on class-specific shared causal paths, the pruned model may preserve the target class prediction while producing lower logits than the original model. Such behavior can become problematic in settings where calibrated confidence matters. In open-set or out-of-distribution recognition, predictions below a confidence threshold may be treated as unknown, making it difficult to distinguish between the target class and unseen classes despite preserving the same decision. Similarly, in calibration-sensitive domains such as medical diagnosis, a confidence reduction (e.g., from 95% to 52%) may trigger unnecessary downstream procedures even when the predicted class remains correct. Addressing these limitations requires extending causal criteria beyond discrete decision preservation toward formulations that also account for continuous confidence preservation. Developing principled causal definitions for continuous-valued transformer behaviors therefore remains an important direction for future work.

Extending Causal Path Tracing to Larger Models. We acknowledge that our experiments were conducted on relatively smaller models compared to state-of-the-art architectures. While our approach substantially reduces the search space over all possible path combinations, scaling to larger models remains challenging, as the number of attention heads increases and the resulting search—although exhibiting polynomial behavior in practice—may still incur prohibitively high computational cost. However, we emphasize that prior work has not demonstrated path-level causal analysis that is both reliable and feasible even for small models. In this sense, our work represents a necessary first step toward practical causal path tracing, establishing a baseline where such analysis is shown to be achievable at all. As a direction for future work, we aim to extend the current block-wise search strategy to more global, model-level schemes that can further accelerate causal path tracing, enabling application to larger-scale transformer architectures.

Extending Causal Path Tracing to Multi-token Outputs. Our current approach, in line with prior work, focuses on tracing internal causal paths for a single model decision. Extending this framework to multi-token output settings introduces several challenges. In particular, independently tracing causal paths for each decision can lead to interpretative redundancy and significant computational inefficiency. As a direction for future work, we aim to address this challenge by viewing the sequence of decisions—particularly in autoregressive language models—as a single unified causal graph. We believe that this perspective enables more efficient causal path tracing and facilitates disentanglement of internal mechanisms shared across multiple decisions.

Downstream Applications and Future Impact. The primary goal of this work is to establish a principled causal path tracing framework for transformer models, rather than to demonstrate immediate downstream performance gains. However, based on the empirical findings presented in this paper, we believe several promising application directions are suggested. First, the observation that repeatedly induced hallucinations give rise to shared causal paths suggests a potential avenue for model debugging, where internal structures responsible for systematic failures could be identified and corrected. Second, beyond demonstrating class-wise shared causal paths, our framework may enable model pruning by preferentially preserving potential task-specific shared causal structures while removing redundant internal computations. More broadly, if future work succeeds in characterizing how causal paths emerge during training, this framework could inform resource-efficient training strategies, enabling models to be trained in a task-adaptive manner with reduced data or computational requirements. Exploring these directions remains an important and open area for future research.

While our framework can improve transparency and debugging of transformer internals, it may also enable targeted manipulation of decision-relevant internal pathways. In particular, sparse intervention on causal paths could potentially be misused to alter model behavior in unintended or deceptive ways. We therefore emphasize that such methods should be studied and deployed with appropriate safeguards and careful evaluation, especially in safety-critical applications.

C Why Causal Referencing Is Necessary: An Illustrative Example

Example 1. *Consider a two-block transformer in which two node sets $V_1^{(2)}$ and $V_2^{(2)}$ in the final block have already been identified as causal node sets, respectively. In this case, whether a node set $V_1^{(1)}$ in the first block is causal may depend on whether it is evaluated jointly with $V_1^{(2)}$, $V_2^{(2)}$, or both. That is, the causal status of $V_1^{(1)}$ can vary depending on which downstream path the internal information propagates through; in other words, some combinations may be sufficient for the decision, while others may not.*

D Why We Use Token Resampling as an Intervention

A central challenge in applying causal theory to transformer internals is that the notion of an intervention in formal causal models does not directly translate to neural representations. In structural causal models, an intervention assumes precise control over variables and their causal mechanisms. However, in transformers, identifying an operation that faithfully satisfies such assumptions is fundamentally difficult. To rigorously guarantee an intervention on internal representations, one would in principle need complete knowledge of the representation manifold, the full internal computational state, and the exact transformation relationship between inputs and all downstream representations.

As a result, many prior studies on transformer internals employ various intervention strategies without formally characterizing what properties those interventions should satisfy from a causal perspective. Rather than claiming that any practical intervention perfectly realizes the idealized intervention of causal theory, we instead aim to operationalize what properties an intervention should satisfy to be sufficiently reliable for tracing causal paths in transformers. To this end, we hypothesize a set of conditions that an intervention should satisfy for operational causal isolation, and provide theoretical toy examples supporting why our chosen intervention strategy satisfies these conditions more appropriately than commonly used alternatives. This discussion is intended primarily as an operational motivation rather than a core theoretical claim.

Definition 6 (Operationally Sufficient Intervention)**.** *In a transformer, we say that replacing the values of a node set V is an **operationally sufficient intervention** on $\mathcal{G}$ for causal isolation if the following conditions are satisfied:*

- *(a) **Causal Structural Isomorphism**: The replacement must admit a one-to-one correspondence with an intervened graph $\mathcal{G}$ where edges are removed. If separately replacing the values of multiple node sets induces the same intervened graph structure, causal source of the decision cannot be uniquely identified under the intervention.*
- *(b) **Causal Edge Validity**: Under such replacement on V, for every child node, there must exist at least one parent node set identifiable as causal. This follows from the fact that, under the no latent confounders assumption and the directed acyclic structure, any change in a child node's value must originate from at least one of its parents.*

(c) ***Intervention Controllability****: The values of a node set V must be replaced with a representation lying on the manifold of the layer where the node set resides. If the replaced representation lies off the manifold, confounding effects from artifacts or unknown factors outside the model's in-distribution may prevent causal source of the decision from being uniquely identified.*

Among possible intervention strategies within transformers, a straightforward approach is to add noise directly to the target node (DIRECT NOISE). However, such perturbation does not guarantee satisfaction of Condition (a) under our framework. Another commonly used strategy, as in prior work such as [6], involves forwarding a noise-perturbed token embedding through the model to obtain a corrupted version of the target node (NOISE TOKEN); however, this does not meet Condition (c) in our setting. A naive alternative, such as zero-masking (ZERO MASK), is also insufficient in this context, as it does not preserve Condition (b).

Since these strategies do not simultaneously satisfy the required conditions in our setting, we instead adopt an intervention method that replaces the target node by resampling from alternative token embeddings (TOKEN RESAMPLING). This approach is shown to satisfy all three conditions for a sufficient intervention.

To illustrate why TOKEN RESAMPLING constitutes a sufficient intervention in our framework, we consider a simplified transformer architecture with a single block and a single attention head; thus, the block input z_{ib} corresponds to the original token embedding z_{token}. The block output can then be decomposed into four additive path terms, as described in Equation (2): the residual only path node (v_1), the Attention only path node (v_2), Attention+MLP path node (v_3), and MLP only path node (v_4).

D.1 For Definition (a) (Causal Structural Isomorphism)

Consider two intervention scenarios where either v_1 or v_4 is replaced. Under TOKEN RESAMPLING, we substitute the original token embedding z_{token} with a different in-vocabulary embedding z'_{token}. This substitution affects both v_1 and v_4, as they both contain z_{token} (i.e., z_{ib}) as their root variable. Importantly, whether z_{token} or z'_{token} appears in v_1 or v_4 leads to distinct expressions for the block output. By these expressions, the resulting interventional graphs remain distinguishable across scenarios. This one-to-one correspondence can be observed in this toy example and extends to more general settings, which supports satisfaction of Definition (a).

In contrast, DIRECT NOISE adds the same Gaussian noise directly to multiple nodes (e.g., v_1 or v_4), which leads to overlapping outcomes across different intervention scenarios. This is due to the additive nature of the block output, where all path node contributions are combined through addition. As a result, the one-to-one mapping between interventional graphs and their mathematical representations is no longer preserved, and Definition (a) is not satisfied in this case.

D.2 For Definition (b) (Causal Edge Validity)

Consider an intervention scenario where all path nodes are replaced. Under ZERO MASKING, each component is zeroed out, which causes the model output to collapse to a constant vector (i.e., the block output $z_{ob} = 0$), resulting in a fixed decision $y = b_{cls}$, where b_{cls} denotes the classifier's bias term. In this case, the intervened decision becomes entirely independent of the original input, relying solely on the classifier bias. As a result, there exists at least one decision for which the causal necessity condition cannot be satisfied for certain decisions, and thus does not satisfy Definition (b).

In contrast, TOKEN RESAMPLING prevents the decision from collapsing to the classifier bias b_{cls} by replacing the original token embedding z_{token} with a different in-vocabulary embedding z'_{token}, ensuring that the causal necessity condition can still be satisfied, thereby supporting Definition (b).

D.3 For Definition (c) (Intervention Controllability)

Consider an intervention scenario where v_4 is replaced. TOKEN RESAMPLING replaces a representation with another token embedding from the model's learned vocabulary. Because these embeddings originate from the model itself, the resulting representations remain on the model's manifold, ensuring that interventions stay within the model's natural distribution, which is consistent with Definition (c).

In contrast, NOISE TOKEN replaces z_{token} with $z_{token} + z_{noise}$, where z_{noise} is the noise. As this composite input is passed through the MLP in v_4, the noise term is transformed by learned parameters, producing a representation that is not guaranteed to lie on the model's manifold. As this noisy signal propagates through subsequent layers, it may lead to out-of-distribution representations, and thus Definition (c) is not ensured in this setting.

E Practical Intervention Procedure

We now describe how the intervention is practically performed in our tracing framework.

First, alternative token sequences are generated by randomly resampling tokens from the model vocabulary while excluding the original input tokens. The resampled sequence preserves the same sequence length as the original input. This process is repeated with different random seeds until 100 resampled cases are obtained whose resulting decisions differ from the original model decision.

Next, each resampled token sequence is independently forwarded through the transformer, and the representation produced at every path node is stored. These stored representations serve as candidate interventional replacements for later tracing.

More specifically, in Equation (2), the additive terms grouped as path nodes are replaced, during intervention, by the representations obtained at the corresponding locations from the resampled token sequences. That is, for a selected path node, its value is substituted with the representation produced at the same location under the resampled input, while all remaining non-intervened terms are left unchanged from the original forward computation (including the terms such as D_α and D_β within that path node term, which are likewise replaced by their corresponding values from the resampled forward pass).

When evaluating a target node set, the stored representations from the 100 resampled cases are substituted into the corresponding target location while the remaining computation is kept unchanged. This produces 100 intervened outputs, each represented as a post-softmax probability vector over the vocabulary. The resulting 100 probability vectors are then averaged into a single aggregated probability vector, from which the final decision is determined by selecting the predicted index.

This intervention procedure is particularly useful for causal tracing because the collected resampled cases are constrained to produce decisions different from the original prediction. As a result, intervening on the complete set of parent representations connected to a downstream node alters the decision behavior under the intervention distribution. Consequently, the downstream node remains causally attributable to some subset of those parent representations under the referencing framework. This property operationally supports why TOKEN RESAMPLING is effective for preserving causal identifiability during tracing.

F Full Derivation of Unfolding Transformer Block

We begin with the following equation, Equation (1):

$$
\begin{aligned}
\left[\, [z_q^{(h)}]_{h=1}^{H};\ [z_k^{(h)}]_{h=1}^{H};\ [z_v^{(h)}]_{h=1}^{H} \,\right] &= \mathtt{L}_{ia}(\mathtt{LN}_1(z_{ib})), \\
z_{oa} &= \mathtt{L}_{oa}([\,\mathrm{softmax}(z_q^{(h)} z_k^{(h)\top}/\sqrt{d_h}) z_v^{(h)}\,]_{h=1}^{H}), \\
z_{im} &= z_{ib} + z_{oa}, \\
z_{om} &= \mathtt{L}_{om}\left(\phi(\mathtt{L}_{im}(\mathtt{LN}_2(z_{im})))\right), \\
z_{ob} &= z_{im} + z_{om},
\end{aligned} \tag{5}
$$

where $z_{ib}, z_{oa}, z_{im}, z_{om}, z_{ob} \in \mathbb{R}^{T\times d_m}$ and $z_q^{(h)}, z_k^{(h)}, z_v^{(h)} \in \mathbb{R}^{T\times d_h}$, where T denotes the number of tokens, d_m the model dimension, H the number of heads, and $d_h = {}^{d_m}\!/_{H}$. Here, this can be unfolded into $2H + 2$ paths, as follows:

$$
z_{ob} = z_{im} + z_{om} = \underbrace{z_{ib}}_{\text{Residual Only "1 } \textit{Path}\text{"}} + z_{oa} + z_{om}.
$$

$$
\begin{aligned}
z_{oa} = \; & \underbrace{(\sum_{h=1}^{H} z_q^{(h)} z_k^{(h)\top} \odot D_\alpha z_v^{(h)} W_{oa}^\top)}_{\substack{\text{Attention Only} \\ \text{(i.e., Attention \textit{params.} with the block input } z_{ib}, \\ \text{as contained in } z_q^{(h)}, z_k^{(h)}, z_v^{(h)})}} + \underbrace{b_{oa}}_{\substack{\text{Bias Terms from Attention} = H \cdot b_{\text{attn}} \\ \text{(i.e., Attention \textit{params.} only)}}}, \\
= & \underbrace{(\sum_{h=1}^{H} z_q^{(h)} z_k^{(h)\top} \odot D_\alpha z_v^{(h)} W_{oa}^\top + b_{\text{attn}})}_{\text{Attention Only ``}H \textit{ Paths}\text{''}}. \\
z_{om} = \; & \underbrace{z_{oa} W_{ln_2}^\top W_{im}^\top \odot D_\beta W_{om}^\top}_{\substack{\text{Attention+MLP} \\ \text{(i.e., MLP \textit{params.} with the attention output } z_{oa})}} \\
& + \underbrace{z_{ib} W_{ln_2}^\top W_{im}^\top \odot D_\beta W_{om}^\top}_{\substack{\text{MLP Only} \\ \text{(i.e., MLP \textit{params.} with the block input } z_{ib})}} + \underbrace{b_{ln_2} W_{im}^\top \odot D_\beta W_{om}^\top + b_{im} \odot D_\beta W_{om}^\top + b_{om}}_{\substack{\text{Bias Terms from MLP} = H \cdot b_{\text{attn+mlp}} + b_{\text{mlp}} \\ \text{(i.e., MLP \textit{params.} only)}}}, \\
= & \left((\sum_{h=1}^{H} z_q^{(h)} z_k^{(h)\top} \odot D_\alpha z_v^{(h)} W_{oa}^\top) + b_{oa} \right) W_{ln_2}^\top W_{im}^\top \odot D_\beta W_{om}^\top \\
& + z_{ib} W_{ln_2}^\top W_{im}^\top \odot D_\beta W_{om}^\top + H \cdot b_{\text{attn+mlp}} + b_{\text{mlp}}, \\
= & (\sum_{h=1}^{H} z_q^{(h)} z_k^{(h)\top} \odot D_\alpha z_v^{(h)} W_{oa}^\top W_{ln_2}^\top W_{im}^\top \odot D_\beta W_{om}^\top) + b_{oa} W_{ln_2}^\top W_{im}^\top \odot D_\beta W_{om}^\top \\
& + z_{ib} W_{ln_2}^\top W_{im}^\top \odot D_\beta W_{om}^\top + H \cdot b_{\text{attn+mlp}} + b_{\text{mlp}}, \\
= & \underbrace{(\sum_{h=1}^{H} z_q^{(h)} z_k^{(h)\top} \odot D_\alpha z_v^{(h)} W_{oa}^\top W_{ln_2}^\top W_{im}^\top \odot D_\beta W_{om}^\top + \frac{b_{oa} W_{ln_2}^\top W_{im}^\top \odot D_\beta W_{om}^\top}{H} + b_{\text{attn+mlp}})}_{\text{Attention+MLP ``}H \textit{ Paths}\text{''}} \\
& + \underbrace{z_{ib} W_{ln_2}^\top W_{im}^\top \odot D_\beta W_{om}^\top + b_{\text{mlp}}}_{\text{MLP Only ``1 } \textit{Path}\text{''}}. \qquad (6)
\end{aligned}
$$

G Proofs

G.1 Proof of Theorem 1

Lemma 1 (Expected Number of Subset Evaluations for Theorem 1)**.** *Consider a minimality-based subset search over n nodes, where each subset is independently selected as a causal node set with probability p under an abstract probabilistic model. Then, the expected number of subset evaluations over all subsets is bounded by:*

$$
n + (1-p) \times \sum_{s=2}^{n} \max \left(0, \binom{n}{s} + \sum_{i=1}^{s-1} \sum_{m=1}^{\lfloor p\binom{n}{i} \rfloor} (-1)^m \binom{p\binom{n}{i}}{m} \binom{n-mi}{s-mi} \right).
$$

Proof. At step $s = 1$, there are $\binom{n}{1} = n$ singleton subsets, all of which must be evaluated, resulting in exactly n evaluations.

For each step $s \geq 2$, the number of candidate subsets of size s is $\binom{n}{s}$. A subset of size s is pruned if it contains any previously selected subset of size $i < s$. Thus, the expected number of subsets to be evaluated at step s is the total number of candidates minus the expected number of pruned subsets:

$$
\binom{n}{s} \; - \; \mathbb{E}[\text{number of pruned subsets at step } s].
$$

To estimate the pruning term, note that the number of subsets of size i is $\binom{n}{i}$, and each is independently selected with probability p. For any such subset of size $i < s$, the number of supersets of size s that include it is $\binom{n-i}{s-i}$, as the remaining $s - i$ nodes must be selected from the $n - i$ nodes not in the subset. Therefore, each selected subset of size i contributes in expectation $p \cdot \binom{n-i}{s-i}$ pruned subsets at step s. Multiplying by the total number $\binom{n}{i}$ of such subsets yields the expected number of pruned subsets due to size-i selections:

$$p\binom{n}{i}\binom{n-i}{s-i}.$$

Summing over all $1 \leq i < s$, the total expected number of pruned subsets at step s is:

$$\sum_{i=1}^{s-1} p\binom{n}{i}\binom{n-i}{s-i}. \tag{7}$$

Equation (7) assumes that pruning effects from different subsets are disjoint. However, a subset of size s may be pruned by multiple selected subsets of size i, which results in overcounting. To correct for this, we apply the inclusion–exclusion principle. Equation (7) corresponds to the first-order term of this expansion, and is fully incorporated into the more general form below. For instance, when two subsets of size i are both selected and jointly included in a size-s subset, that subset is overcounted in the previous estimate and must be subtracted once. The number of such expected double-overlaps is:

$$-\sum_{i=1}^{s-1}\binom{\lfloor p\binom{n}{i}\rfloor}{2}\binom{n-2i}{s-2i}.$$

This logic generalizes to arbitrary overlap levels. When m mutually disjoint subsets of size i are selected and jointly included in a size-s subset, the corresponding correction term alternates in sign according to the inclusion–exclusion principle, and takes the form:

$$(-1)^m\binom{\lfloor p\binom{n}{i}\rfloor}{m}\binom{n-mi}{s-mi}.$$

Summing over all $m \geq 1$, the total expected number of pruned subsets at step s, with successive overcount corrections added and subtracted depending on the overlap level, is:

$$\sum_{i=1}^{s-1}\sum_{m=1}^{\lfloor p\binom{n}{i}\rfloor}(-1)^m\binom{\lfloor p\binom{n}{i}\rfloor}{m}\binom{n-mi}{s-mi}. \tag{8}$$

To obtain the total expected number of subset evaluations over all steps, we sum the unpruned subsets across all sizes. At step $s = 1$, all n singleton subsets are evaluated. For $s \geq 2$, each subset of size s survives pruning with probability $1 - p$, and only if it has not been eliminated due to overlap with smaller selected subsets. Thus, the expected number of evaluations at step s is:

$$(1-p)\cdot\left(\binom{n}{s} + \sum_{i=1}^{s-1}\sum_{m=1}^{\lfloor p\binom{n}{i}\rfloor}(-1)^m\binom{\lfloor p\binom{n}{i}\rfloor}{m}\binom{n-mi}{s-mi}\right).$$

The inner summation captures the pruning corrections from the inclusion–exclusion expansion in Equation (8). To ensure non-negativity of the expected number at each step, we take a maximum with zero.

Adding this to the evaluations from $s = 1$, the total expected number of subset evaluations is:

$$n + (1-p)\sum_{s=2}^{n}\max\left(0, \binom{n}{s} + \sum_{i=1}^{s-1}\sum_{m=1}^{\lfloor p\binom{n}{i}\rfloor}(-1)^m\binom{\lfloor p\binom{n}{i}\rfloor}{m}\binom{n-mi}{s-mi}\right).$$

□

Theorem 1 (Expected Number of Subset Evaluations under Minimality Search)**.** *Consider a minimality-based subset search over n nodes, where each subset is independently selected as a causal node set with probability p under an abstract probabilistic model. Then, based on Lemma 1, the expected number of evaluated subsets is upper-bounded approximately by:*

$$O\left(n^{\lfloor \log_2(\frac{1}{p}+2)\rfloor}\right).$$

Proof. From Lemma 1, the expected number of subset evaluations is:

$$n + (1-p)\sum_{s=2}^{n} \max\left(0, \binom{n}{s} + \sum_{i=1}^{s-1}\sum_{m=1}^{\lfloor p\binom{n}{i}\rfloor} (-1)^m \binom{\lfloor p\binom{n}{i}\rfloor}{m}\binom{n-mi}{s-mi}\right). \tag{9}$$

We approximate the pruning term by keeping only the $m = 1$ intersection term in the inclusion-exclusion sum, discarding the rest ($m \geq 2$), which largely cancel out due to alternating signs. Here, $\binom{\lfloor p\binom{n}{i}\rfloor}{m}$ reduces to $\lfloor p\binom{n}{i}\rfloor$, where the floor was introduced to ensure the argument of the binomial coefficient is an integer; since this is no longer needed, we rewrite it as $p\binom{n}{i}$.

$$\binom{n}{s} - \sum_{i=1}^{s-1} p\binom{n}{i}\binom{n-i}{s-i}.$$

Using the fact that $\binom{n}{i}\binom{n-i}{s-i} = \binom{n}{s}\binom{s}{i}$ and $\sum_{i=1}^{s-1}\binom{s}{i} = 2^s - 2$, we simplify the above as:

$$\binom{n}{s}\left(1 - p(2^s - 2)\right).$$

Substituting this into Equation (9), we obtain the following:

$$n + (1-p)\sum_{s=2}^{n} \max\left(0, \binom{n}{s}(1 - p(2^s - 2))\right). \tag{10}$$

A subset of size s contributes to the sum ($\sum_{s=2}^{n}$) only if the term with pruning factor is positive, i.e., the term $(1 - p(2^s - 2))$ applied to the positive binomial term $\binom{n}{s}$ is greater than zero:

$$1 - p(2^s - 2) > 0 \quad \leftrightarrow \quad s < \log_2\left(\frac{1}{p} + 2\right).$$

Here, let $s^* := \log_2\left(\frac{1}{p} + 2\right)$ be the largest such s that satisfies this condition, where $\lfloor s^* \rfloor$ serves as the integer upper limit of the sum. Then, Equation (10) reduces to:

$$n + (1-p)\sum_{s=2}^{\lfloor s^*\rfloor} \binom{n}{s}(1 - p(2^s - 2)).$$

Since each binomial coefficient satisfies $\binom{n}{s} = O(n^s)$ and the term $(1 - p(2^s - 2))$ acts as a constant for each fixed s, the summation is upper-bounded as

$$\sum_{s=2}^{\lfloor s^*\rfloor} \binom{n}{s}(1 - p(2^s - 2)) = O(n^2 + \cdots + n^{\lfloor s^*\rfloor}) = O(n^{\lfloor s^*\rfloor}).$$

Thus, the total number of evaluations grows approximately as

$$n + (1-p)\cdot O(n^{\lfloor s^*\rfloor}) = O\left(n^{\lfloor s^*\rfloor}\right) = O\left(n^{\lfloor \log_2(\frac{1}{p}+2)\rfloor}\right).$$

□

G.2 Proof of Theorem 2

Theorem 2 (Causal Union Reference Reliability)**.** *Consider a minimality-based subset search over n nodes, where each subset is independently selected as a causal node set with probability p under an abstract probabilistic model. Suppose that a collection of such sets, $V_{out}^{(j+1)} = \{V_i^{(j+1)}\}_{i=1}^{k}$, is identified as causal in the $(j+1)$-th block. Their union, denoted as $P = \bigcup_{i=1}^{k} V_i^{(j+1)}$, serves as the causal subpath reference for the minimality-based subset search in the j-th block. Let s_{avg} denote the average size of the k causal node sets in $V_{out}^{(j+1)}$. Then, the reliability of the resulting causal node set obtained using P is given by:*

$$p + (1-p)\left(1 - \left(1 - \frac{s_{avg}}{n}\right)^k\right)^n \to 1 \tag{11}$$

Proof. To evaluate the reliability of using the union $\bigcup_{i=1}^{k} V_i^{(j+1)}$ as the causal subpath reference P during the search in the j-th block, we consider the following two cases under which this strategy ensures a reliable outcome. First, P itself could have been selected as a causal node set during the search in the $(j+1)$-th block, independently with probability p, even though it was not explicitly selected because its subsets had already been included. Second, if P was not selected in this way, it must equal the full set of path nodes in the $(j+1)$-th block. The total probability of success across these two cases is given by:

$$p + (1-p) \cdot \hat{p}, \tag{12}$$

where $\hat{p}$ denotes the probability of the second case. To compute $\hat{p}$, note that the probability that a single causal node set in $V_{\text{out}}^{(j+1)}$, with average size s_{avg}, contains a given node among the n path nodes in the j-th block is:

$$\frac{s_{\text{avg}}}{n}.$$

Accordingly, the probability that at least one of the k causal node sets contains a given node is:

$$1 - \left(1 - \frac{s_{\text{avg}}}{n}\right)^k.$$

Extending this, the probability that the k causal node sets collectively cover all n path nodes, i.e., that P equals the full set of path nodes in the j-th block, is:

$$\hat{p} = \left(1 - \left(1 - \frac{s_{\text{avg}}}{n}\right)^k\right)^n.$$

Substituting this into Equation (12), the reliability can be expressed as follows:

$$p + (1-p) \cdot \left(1 - \left(1 - \frac{s_{\text{avg}}}{n}\right)^k\right)^n.$$

To analyze convergence, define $x = \frac{s_{\text{avg}}}{n}$. Using the inequality:

$$(1-x) \leq e^{-x},$$

we obtain:

$$\left(1 - (1-x)^k\right)^n \geq \left(1 - e^{-xk}\right)^n.$$

Substituting back $x = \frac{s_{\text{avg}}}{n}$, the lower bound for $\hat{p}$ becomes:

$$\hat{p} \geq \left(1 - e^{-k \cdot \frac{s_{\text{avg}}}{n}}\right)^n.$$

Here, k is equal to the number of subset evaluations selected with probability p from the minimality-based subset search. As n grows large, k asymptotically follows the evaluation bound shown in Lemma 1, which converges to n^{s^*}. Therefore, k can be approximated as:

$$k \approx p \cdot n^{s^*} = p \cdot n^{\log_2\left(\frac{1}{p}+2\right)}.$$

Thus, the reliability is lower bounded as:

$$\begin{aligned} p + (1-p) \cdot \left(1 - \left(1 - \frac{s_{\text{avg}}}{n}\right)^k\right)^n &\geq p + (1-p) \cdot \left(1 - e^{-k \cdot \frac{s_{\text{avg}}}{n}}\right)^n \\ &\approx p + (1-p) \cdot \left(1 - e^{-p \cdot n^{s^*} \cdot \frac{s_{\text{avg}}}{n}}\right)^n \\ &= p + (1-p) \cdot \left(1 - e^{-p \cdot s_{\text{avg}} \cdot n^{s^*-1}}\right)^n. \end{aligned}$$

Here, $s^* - 1$ can be rewritten as follows:

$$s^* - 1 = \log_2\left(\frac{1}{p} + 2\right) - 1 = \log_2\left(\frac{1}{2p} + 1\right).$$

Since $\frac{1}{2p} + 1 > 1$ for all $p \in (0, 1]$, we have $s^* - 1 > 0$. Consequently, $p \cdot s_{\text{avg}} \cdot n^{s^*-1} \to \infty$ as n grows, which gives:

$$e^{-p \cdot s_{\text{avg}} \cdot n^{s^*-1}} \to 0 \quad \Rightarrow \quad \left(1 - e^{-p \cdot s_{\text{avg}} \cdot n^{s^*-1}}\right)^n \to 1.$$

Then, the lower bound satisfies:

$$p + (1-p) \cdot \left(1 - e^{-p \cdot s_{\text{avg}} \cdot n^{s^*-1}}\right)^n = p + (1-p) \cdot 1 = 1.$$

Since reliability is a probability upper bounded by 1, the lower bound converging to 1 implies that the reliability itself converges to 1 as n grows:

$$p + (1-p)\left(1 - \left(1 - \frac{s_{\text{avg}}}{n}\right)^k\right)^n \to 1.$$

In conclusion, considering the two cases above, the union of the k causal node sets identified in the $(j+1)$-th block serves as a reliable causal subpath reference P for the j-th block as n grows.

□

H Implementation Details and Additional Results

H.1 Common Experimental Setup

The code is publicly available at `https://github.com/Jo-won/CausalPathTracing`. Experiments were conducted using an NVIDIA Quadro RTX 6000 GPU with 24 GB of memory, except for the scalability experiments in Appendix Section H.8, which used an NVIDIA RTX PRO 5000 GPU with 48 GB of memory to accommodate larger models. We used PyTorch 2.1.0 and CUDA 11.8. For three language models (GPT2-xs[1], Pythia-14m[2], and Pythia-1b[3]) and two vision models (ViT-tiny and DeiT-tiny[4]), the internal decision paths were traced while keeping all pretrained parameters frozen. Note that for the OFFICEHOME dataset, only the classifier was fine-tuned to match the number of classes. In the case of Pythia, since it adopts a parallel residual structure where the block input is directly fed into both the attention and MLP components, unfolding is adapted accordingly to reflect this architecture; as a result, each block produces $H + 2$ paths instead of $2H + 2$. Since a decision path is considered meaningful only if the model produces the correct output for a given input, we sampled a subset of correctly predicted examples from the evaluation set and used the same subset across all methods to ensure a fair comparison.

When using our proposed method with TOKEN RESAMPLING, to prevent incorrect interventions caused by coincidentally selecting token embeddings similar to the original input, we sampled 100 different resampling batches and computed the causal evaluation as the average output across these interventions. Clean path-node activations were computed using the gating values D_α and D_β from the original forward pass. Replacement path-node activations were computed using the corresponding gating values from each resampled forward pass. Following the identification of causal paths, we utilized their union for further analysis, as justified by Theorem 2.

[1] `https://huggingface.co/AlgorithmicResearchGroup/gpt2-xs`
[2] `https://huggingface.co/EleutherAI/pythia-14m`
[3] `https://huggingface.co/EleutherAI/pythia-1b`
[4] `https://github.com/huggingface/pytorch-image-models`

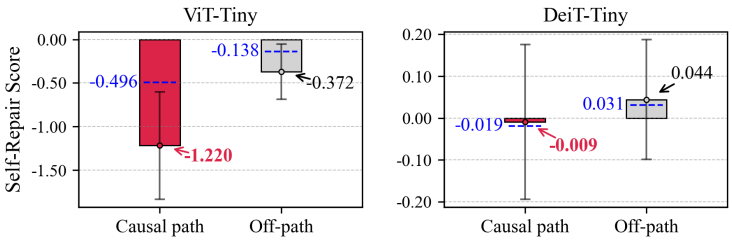


Figure A: **Self-repair scores on causal path vs. off-path components in vision models.** Each bar shows the mean (dot with arrow) and standard deviation (error bar); medians are shown as blue dashed lines. Lower scores indicate less self-repair. Results are averaged over IMAGENET and OFFICEHOME.

H.2 How Self-Repair Is Computed

Building on the notion of self-repair introduced by the prior work [17], we define the self-repair score received by each attention head in a given layer as:

$$\text{self-repair} = \Delta logit - \Delta DE_{head}. \tag{13}$$

Here, $\Delta logit$ measures the change in the logit corresponding to the original decision after ablating the attention head. ΔDE_{head} measures the direct effect of the head, i.e., its original contribution to the decision logit without self-repair. The direct effect is obtained by directly multiplying the classifier weight to the target head.

Specifically, the final residual stream, i.e., the classifier input, contains the sum of outputs from all heads across all layers. Accordingly, the contribution of each head can be directly approximated by projecting its output onto the classifier weight vector; this yields the direct effect of that head, ΔDE_{head}. Furthermore, following the assumption in the prior work [17] that an ablated head contributes zero to the decision, we compute $\Delta DE_{head} = -DE_{head}$.

H.3 What Self-Repair Means in Our Context

Self-repair, also referred to as backup behavior or a backup mechanism, serves a functional role in maintaining the model's behavior. In this section, we aim to clarify both its meaning and its implications in relation to causal paths.

In our context, the self-repair score reflects how much a target node receives backup from other nodes, thereby indicating its role as a backup receiver. In other words, there are two roles in a backup mechanism during ablation: a giver node, which supplies the compensatory signal, and a receiver node, which is affected by it. The self-repair score we measure captures the extent to which a node functions as a receiver in this process.

Specifically, for a node to act as a giver, it must be capable of supporting the decision itself; its contribution to the decision must be high. Given that the total logit for the decision is a fixed quantity, greater contribution by givers implies lesser contribution from receivers. In this sense, giver nodes involved in backup mechanisms may be seen as indicative of the causal mechanism itself. Therefore, if the discovered paths contain many receiver nodes, their self-repair scores will be high, indicating a lower contribution to the decision compared to other paths composed primarily of giver nodes; that is, such discovered paths are likely less causal. In contrast, if receiver nodes mostly lie outside the discovered paths, their self-repair scores within the paths will be lower than those outside, suggesting that the discovered paths contribute more directly to the decision than nodes outside the paths. Taken together, a low self-repair score indicates that the discovered path contributes more directly to the decision than components outside it, thereby serving as evidence of its causal nature.

H.4 Self-repair Analysis in Vision Models

Setup As described in Section H.2, we measure the self-repair score of each attention head across layers. A head is considered to be on the causal path if it is included in any causal path at least once; otherwise, it is classified as off-path.

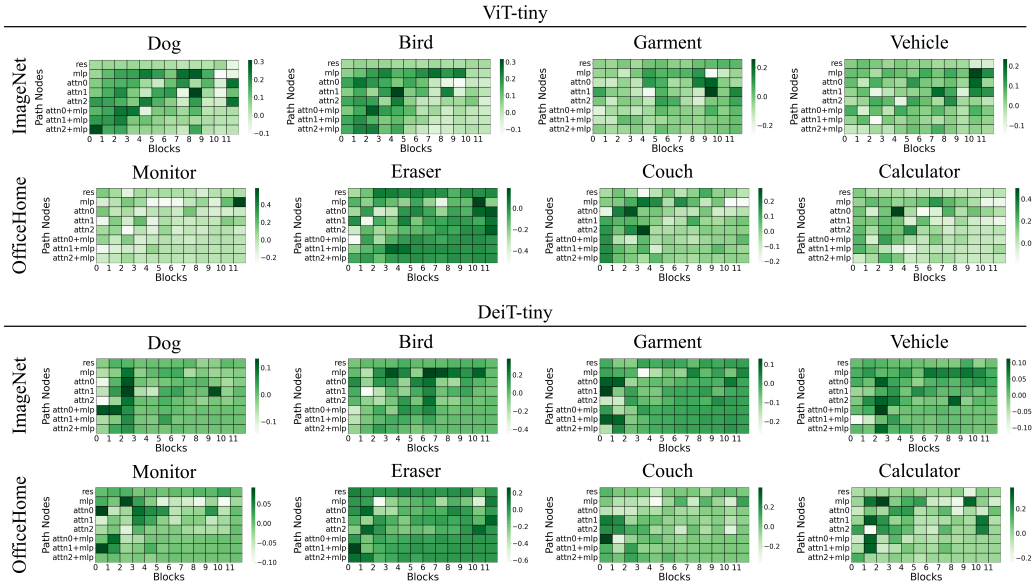


Figure B: **Class-specific causal path patterns.** The class-wise difference in average causal path ratios between a target class and all classes, highlighting class-specific paths. Here, res, mlp, and attn# indicate residual, MLP, and attention paths from head #, respectively.

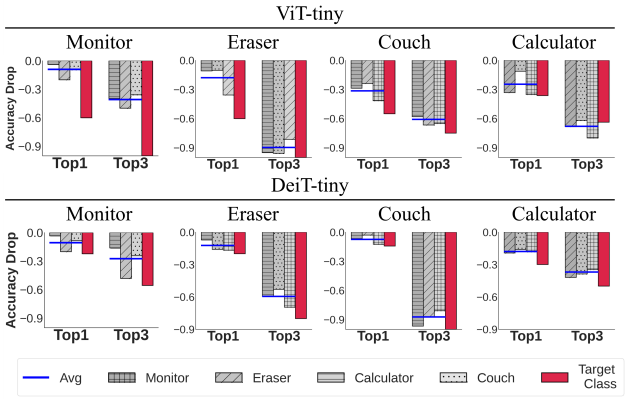


Figure C: **Causal paths uniquely activated for specific classes on OFFICEHOME.** Accuracy drop when ablating the most class-specific path, showing selective reliance by each class.

Results Extending the findings from the main text on language models, we examine whether the same pattern holds in vision models. As shown in Figure A, although self-repair behavior is inherently noisy, as previously reported, we still observe consistently lower self-repair scores on components included in the causal path. This suggests that, even in vision models, the identified causal paths correspond to components that play an important and non-redundant role in the decision-making process of the model.

H.5 Class-specific Path Discovery

Setup We conducted experiments by removing the largest activated causal subpath (Top-1 in Figures C and 4) or the top three largest ones (Top-3 in Figures C and 4) for a given super-class. For IMAGENET, we selected a subset of 1,000 IMAGENET classes and grouped them into four super-classes based on WordNet semantic similarity. The defined superclasses are as follows: **dog**, which includes Maltese, beagle, Saluki, Siberian husky, and golden retriever; **bird (oscines)**, consisting of brambling, robin, jay, and chickadee; **garment**, comprising kimono, cardigan, T-shirt, and sweatshirt; and **vehicle**, which includes cab, minibus, moving van, police van, and school bus. In contrast, in

Method	Language Model	KNOWNS1000			T-REX			Vision Model	IMAGENET			OFFICEHOME		
		Hit. (↑)	$\mathbf{FS}_t$	$\mathbf{PS}_t$	**Hit.** (↑)	$\mathbf{FS}_t$	$\mathbf{PS}_t$		**Hit.** (↑)	$\mathbf{FS}_t$	$\mathbf{PS}_t$	**Hit.** (↑)	$\mathbf{FS}_t$	$\mathbf{PS}_t$
NT_1	GPT2-xs	0.000	0.000	0.071	0.000	0.000	0.071	DeiT-tiny	0.000	0.000	0.125	0.018	0.001	0.125
	Pythia-14m	0.000	0.041	0.167	0.000	0.029	0.167	ViT-tiny	0.000	0.002	0.125	0.014	0.011	0.126
	Pythia-1b	0.000	0.017	0.100	0.000	0.021	0.100	–	–	–	–	–	–	–
$NT_{10\%}$	GPT2-xs	0.000	0.033	0.077	0.000	0.006	0.074	DeiT-tiny	0.000	0.000	0.125	0.018	0.000	0.125
	Pythia-14m	0.000	0.050	0.179	0.000	0.038	0.175	ViT-tiny	0.000	0.000	0.125	0.018	0.001	0.125
	Pythia-1b	0.000	0.060	0.111	0.000	0.063	0.111	–	–	–	–	–	–	–
ET_{all}	GPT2-xs	0.556	0.953	0.750	0.709	0.972	0.702	DeiT-tiny	0.483	0.972	0.740	0.532	0.981	0.760
	Pythia-14m	0.000	0.888	0.750	0.000	0.888	0.667	ViT-tiny	0.285	0.972	0.844	0.405	0.981	0.792
	Pythia-1b	0.000	0.958	0.844	0.000	0.958	0.838	–	–	–	–	–	–	–
ET_{cls}	GPT2-xs	0.556	0.952	0.740	0.631	0.955	0.718	DeiT-tiny	0.441	0.964	0.742	0.315	0.953	0.718
	Pythia-14m	0.104	0.930	0.687	0.201	0.961	0.741	ViT-tiny	0.070	0.929	0.726	0.217	0.899	0.680
	Pythia-1b	0.554	0.988	0.848	0.448	0.991	0.838	–	–	–	–	–	–	–
ET-IG_{all}	GPT2-xs	0.556	0.963	0.774	0.466	0.963	0.726	DeiT-tiny	0.721	0.963	0.813	0.611	0.981	0.750
	Pythia-14m	0.373	0.998	0.917	0.791	0.998	0.917	ViT-tiny	0.021	0.981	0.781	0.175	0.972	0.708
	Pythia-1b	<u>0.976</u>	1.000	0.906	0.970	1.000	0.913	–	–	–	–	–	–	–
ET-IG_{cls}	GPT2-xs	0.611	0.932	0.698	0.534	0.917	0.721	DeiT-tiny	0.593	0.964	0.768	0.397	0.945	0.714
	Pythia-14m	0.269	0.997	0.808	0.458	0.997	0.812	ViT-tiny	0.030	0.911	0.702	0.127	0.895	0.669
	Pythia-1b	0.964	1.000	0.895	0.969	1.000	0.903	–	–	–	–	–	–	–
CPT	GPT2-xs	**1.000**	0.720	0.578	<u>0.990</u>	0.699	0.558	DeiT-tiny	<u>0.968</u>	0.645	0.486	**0.963**	0.662	0.505
	Pythia-14m	0.970	0.956	0.725	**0.992**	0.945	0.696	ViT-tiny	**0.974**	0.779	0.623	<u>0.951</u>	0.687	0.526
	Pythia-1b	0.952	0.689	0.470	0.984	0.658	0.444	–	–	–	–	–	–	–

Table A: **Detailed quantitative results.** Detailed results across language and vision models. The language results are reported on KNOWNS1000 and T-REX; the vision results are reported on IMAGENET and OFFICEHOME. The average performance of each method across models and datasets is reported in Table 3. **Hit.** (hit rate) measures whether the pruned model preserves the original decision. $\mathbf{FS}_t$ denotes the fraction of FLOPs retained by the identified path, and $\mathbf{PS}_t$ denotes the fraction of path nodes retained. Lower values indicate sparser paths.

OFFICEHOME, the visual dissimilarity between classes is too large to form meaningful super-classes, so each class was treated as its own super-class.

Results The activation ratio patterns of the causal paths for each super-class can be seen in Figure B. The stronger the color contrast, the more exclusively the corresponding causal subpath is activated for that specific super-class. Furthermore, similar to the main script results, Figure C shows that in OFFICEHOME, ablating the most activated causal subpath for each super-class leads to a significantly larger accuracy drop for that super-class compared to the average drop across others. This indicates the existence of causal subpaths that are specifically responsible for class-specific decisions. However, when more causal subpaths are ablated, the performance drop becomes more uniform, suggesting that, since each class is treated as its own super-class, the model does not rely heavily on a large number of unique subpaths per class. In other words, critical components for fine-grained decisions appear to be more sparsely encoded within the model.

H.6 Detailed Quantitative Comparisons

The results across models and datasets—those averaged in Table 3 of the main script—are presented in Table A. As previously mentioned, our method uniquely achieves a near-perfect hit rate, reliably identifying the causal path responsible for the original decision.

While some baselines achieve relatively high hit rates in specific settings, they typically do so by retaining a large fraction of the model computation and path nodes. For instance, ET-IG_{all} and ET-IG_{cls} achieve competitive hit rates for Pythia-1b, but their $\mathbf{FS}_t$ and $\mathbf{PS}_t$ values are close to one, indicating that these methods preserve the original decision largely by retaining nearly the entire internal structure. In contrast, CPT consistently achieves near-perfect hit rates while retaining substantially fewer FLOPs and path nodes.

Model	Total Search Space ($2^{n \times D}$)	NT ($\frac{3 \times D}{2^{n \times D}} \ll 0.0001\%$)	ET ($\frac{\binom{(H+2) \times D}{2}}{2^{n \times D}} \ll 0.0001\%$)	CPT ($\approx 100\%$)
GPT2-xs	$2^{14 \times 6}$	0.16sec $\pm$ 0.03	1.80sec $\pm$ 0.02	17632.17sec $\pm$ 2113.36
Pythia-1b	$2^{10 \times 16}$	1.31sec $\pm$ 0.17	8.49sec $\pm$ 0.36	3114.00sec $\pm$ 930.95
Pythia-14m	$2^{6 \times 6}$	0.13sec $\pm$ 0.01	1.76sec $\pm$ 0.01	10.04sec $\pm$ 0.86
ViT-tiny	$2^{8 \times 12}$	0.96sec $\pm$ 0.04	28.07sec $\pm$ 3.05	734.04sec $\pm$ 283.12
DeiT-tiny	$2^{8 \times 12}$	1.00sec $\pm$ 0.04	30.14sec $\pm$ 4.09	564.00sec $\pm$ 86.76

Table B: **Actual wall-clock time of search.** The "Total Search Space" column represents all possible combinations of internal path nodes across blocks, where n is the number of paths per block, D is the number of blocks, and H is the number of heads per block. Here, n is generally $2H + 2$ due to sequential residual connections across attention and MLP, whereas $n = H + 2$ for the Pythia family due to parallel residual connections. The percentages in the column headings indicate each method's coverage of the total search space. For CPT, this denotes coverage in principle; supersets pruned by minimality are skipped without evaluation. NT evaluates attention and MLP outputs independently, resulting in $3 \times D$ search trials. ET considers pairwise combinations among attention heads, MLP, and residual connections—$(H + 2) \times D$ in total—resulting in $\binom{(H+2) \times D}{2}$ search trials.

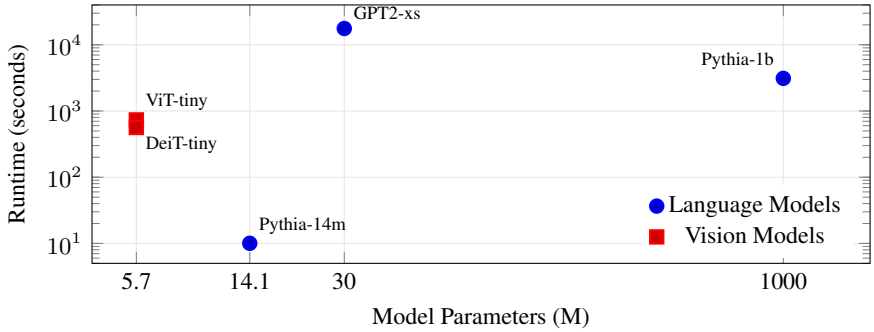


Figure D: **Model size versus runtime of CPT.** Actual wall-clock runtime without a restricted subset-size budget. Both axes use logarithmic scales. Runtime varies with model size, architectural properties, and the resulting subset search space.

This pattern also appears in the vision setting. Although ET-IG$_{\text{all}}$ obtains a relatively high hit rate for DeiT-tiny on IMAGENET, it retains large fractions of FLOPs and path nodes, whereas CPT preserves the original decision with much lower $\mathbf{FS}_t$ and $\mathbf{PS}_t$. These results support our main claim that CPT identifies decision-relevant paths more reliably and compactly than node- or edge-level baselines.

H.7 Actual Wall-Clock Time and Search Space

We aim to explain why exploring all possible path combinations was intractable in prior work, and why achieving tractability (on average) through our method is such a crucial advancement, as evidenced by the actual wall-clock time in Table B.

For each model, every block contains n paths—typically $2H + 2$, as shown in Equation (2)—and the number of possible combinations is 2^n (including the empty set for simplicity). Across D blocks, the total number of possible paths from input to decision amounts to $2^{n \times D}$. This represents the total search space of a exhaustive search strategy required to guarantee reliable path tracing for a decision. The scale of this space is enormous and practically infeasible to exhaustively explore.

Consequently, NT restricts its exploration to combinations of attention and MLP outputs, while ET considers only pairwise combinations, excluding interactions among three or more components. ET-IG further increases the computational cost of ET through integrated gradients, requiring approximately five times longer runtime. Nevertheless, these methods explore only a tiny fraction of the total search space, resulting in substantially shorter runtimes but considerably lower hit rates (Table 3). In contrast, CPT enables more comprehensive path-level causal analysis through minimality-based search and union-based referencing while maintaining high decision preservation. Furthermore, despite its more comprehensive search, CPT requires approximately half the peak GPU memory of ET and ET-IG, excluding model weights, with memory consumption comparable to NT.

Model	Nodes per Block	Blocks	Avg. Causal Set Size	Resolved Blocks (%)	Runtime (min)	Memory (GB) Total / Tracing
Pythia-14m	6	6	2.80	73	0.0	0.17 / 0.05
Pythia-1b	10	16	2.01	97	0.7	2.49 / 0.16
Pythia-2.8b	34	32	1.76	99	23.1	7.00 / 0.81
Pythia-6.9b	34	32	1.68	100	29.3	15.98 / 1.30
Pythia-12b	42	36	1.58	100	90.1	27.08 / 2.07

Table C: **Scalability of CPT under a restricted search budget.** Results on KNOWNS1000 with FP16 inference and a maximum subset size of three ($s \leq 3$). Avg. Causal Set Size denotes the average size of identified minimal causal node sets. Resolved Blocks denotes the percentage of blocks in the final quarter for which a causal node set is found within the search budget. Memory reports total GPU consumption and the additional memory required for tracing (excluding model weights). Unresolved blocks retain their full node sets. All configurations achieve a hit rate of 1.00 under this protocol.

Figure D further examines how CPT's runtime varies with model size. Notably, GPT2-xs (30M parameters) requires approximately 5.7 times longer runtime than Pythia-1b (1B parameters), despite being substantially smaller. This suggests that runtime depends not only on parameter count but also on architectural properties and the resulting subset search space. In particular, GPT2-xs has 14 path nodes per block due to its sequential residual structure, compared with 10 in Pythia-1b with parallel residual connections. These architectural differences may substantially affect the number of subset evaluations required during tracing.

Overall, although CPT requires longer runtime than existing methods, this cost should be considered in light of its substantially more comprehensive exploration of the search space. Given that existing methods examine only a tiny fraction of possible path combinations, these results suggest that CPT achieves practical tractability relative to the scope of its search while providing substantially more reliable causal paths. Nevertheless, its actual runtime remains sensitive to the underlying search-space structure, motivating further improvements in search efficiency for larger models.

H.8 Scalability under a Practical Search Budget

As discussed in Appendix Section B, scaling CPT to larger models remains challenging, as the increasing number of attention heads can lead to prohibitively high computational costs despite its empirical polynomial behavior. Nevertheless, to investigate the potential scalability of CPT, we extend our evaluation to larger Pythia models, ranging from 14M to 12B parameters, under a restricted search budget ($s \leq 3$) on KNOWNS1000. Specifically, we cap the minimality search at a maximum subset size of three. If no minimal causal set is identified within this budget, the corresponding block is treated as the full node set. All experiments are conducted using FP16 inference while keeping the evaluation protocol identical across model scales.

Under this setup, we observe that the average minimal causal-set size consistently decreases as model scale increases ($2.80 \rightarrow 2.01 \rightarrow 1.76 \rightarrow 1.68 \rightarrow 1.58$), indicating that, for the same task, the relevant causal structures remain concentrated in small subsets under the same restricted budget. Also, the additional memory required for tracing remains modest (0.05–2.07 GB).

Furthermore, many downstream applications first localize decision-relevant components and then operate on them (e.g., knowledge editing), typically focusing on the later transformer blocks that are most directly connected to the final prediction. We therefore separately evaluate the final quarter of transformer blocks and find that the proportion of blocks resolved within a maximum subset size of three consistently increases with model scale ($73\% \rightarrow 97\% \rightarrow 99\% \rightarrow 100\% \rightarrow 100\%$). Thus, these results suggest that, when the nodes localized by CPT are used in downstream applications focusing on later blocks, a small search budget may remain practically useful even as model scale increases.

Nevertheless, scaling CPT to substantially larger models remains challenging when the underlying decision requires a large causal node set. Without introducing heuristic approximations, identifying such sets is fundamentally constrained by the exponential worst-case search space. We therefore leave the development of more scalable and principled search strategies as an important direction for future work.

H.9 Comparison with IOI Ground Truth

We highlight the connection between our causal path tracing and the ground-truth circuits manually annotated in prior work [5] for the IOI (Indirect Object Identification) task. Since the IOI task also requires identifying the critical circuits responsible for a model's decision, it is closely related to our approach, and our method achieves high precision on this task: CPT: 0.727, NT_1: 0.146, $\text{NT}_{10\%}$: 0.392, ET_{all}: 0.157, $\text{ET-IG}_{\text{all}}$: 0.154.

Specifically, the ground-truth circuits for IOI were generated using GPT2-small [29]. Because the total search space in GPT2-small is extremely large ($2^{26\times12}$), we evaluated performance on 20 randomly selected samples. Furthermore, since the IOI task does not involve class labels, we exclude ET_{cls} and $\text{ET-IG}_{\text{cls}}$ from this comparison. We also employed a weakened variant of CPT: as this model has a relatively large number of heads ($H = 12, n = 26$), our minimality-based path tracing can get stuck in worst-case traps, resulting in prohibitively long runtimes. To address this, we restricted the search to subsets of size up to three. Even under this constraint, our method shows strong alignment with the IOI ground-truth circuits, as evidenced by its high precision.

Notably, while other methods achieved recall close to 1, they tended to return nearly the entire model as a decision path rather than pinpointing the critical components. For this reason, we regard precision as the more meaningful metric for evaluating the connection to the IOI task.

H.10 Comparison with Gradient-based Attribution Baselines

Baselines. We compare CPT with two gradient-based attribution methods, Gradient (Grad.) [30] and Integrated Gradients (IG) [28]. Grad. selects the node set with the largest attribution score at each block, computed as the product of its clean activation and the gradient of the decision score. IG follows the same selection procedure but integrates gradients along the interpolation path from zero activation to the clean activation. Unlike ET and ET-IG, which use corrupted-to-clean activation differences, Grad. and IG compute attribution relative to zero activation, measuring sensitivity to the clean decision rather than a task-relevant counterfactual change.

Method	**Hit Rate**	**FS_t**	**PS_t**
Grad.	0.141	0.763	0.545
IG	0.165	0.776	0.556
ET_{cls}	0.416	0.963	0.762
$\text{ET-IG}_{\text{cls}}$	0.634	0.974	0.806
CPT	**0.981**	0.778	0.578

Table D: **Comparison with gradient-based attribution baselines.** Hit rate and sparsity metrics on language models.

Results. As shown in Table D, CPT, Grad., and IG retain similarly sized paths, yet Grad. and IG achieve substantially lower hit rates than CPT (0.141 and 0.165 vs. 0.981). These results suggest that gradient-based attribution is more prone to selecting false-positive nodes that do not contribute to forming a sufficient path for reproducing the original decision. This may be because Grad. and IG measure gradient-based sensitivity rather than evaluating intervention effects, potentially assigning high attribution to nodes associated with the decision but not causally required to reproduce it.

ET and ET-IG partially mitigate this limitation by incorporating task-relevant corrupted-to-clean activation differences, achieving higher hit rates than Grad. and IG. Nevertheless, despite retaining substantially larger paths, both methods still achieve lower hit rates than CPT. Taken together, these results suggest that, within the evaluated setting, CPT more selectively identifies decision-relevant nodes by explicitly evaluating necessity and sufficiency.

H.11 Comparison with Path-Level Baselines

We further compare CPT with existing path-level methods, including Accelerated Path Patching (APP) [31] and Path-level Patching-based Tracing (PT), adapted from the path-level method of Hanna et al. [15]. All experiments are conducted on Pythia-1b with KNOWNS1000. As shown in Table E, CPT achieves a higher hit rate with more compact paths than the path-level baselines.

Method	Hit. (↑)	$\mathbf{PS}_t$	Runtime (s)
CPT	**0.952**	0.470	3,114
APP	0.699	0.688	2,413
PT (Top-1)	0.000	0.109	3,114
PT (Top-61)	0.012	0.471	3,114
PT (Top-100)	0.084	0.693	3,114
PT (Top-150)	0.759	0.896	3,114
PT (Top-200)	0.940	0.954	3,114
PT (Top-all)	0.940	0.987	3,114

Table E: **Comparison with path-level baselines.** Hit rate and path node sparsity on Pythia-1b with KNOWNS1000. Runtime is measured in seconds. PT is evaluated under the same runtime budget as CPT; Top-k denotes the union of the k highest-ranked paths.

For APP, we first verified that our implementation reproduces the behavior reported in the original work and then applied it to our evaluation setting. For PT, following Hanna et al. [15], we rank paths using the logit difference between the original decision token and an incorrect token. We adapt its path enumeration into a depth-first search that constructs paths in ascending order of node-set size across blocks. To ensure a fair runtime comparison, PT is terminated after the same runtime budget used by CPT. We then rank the discovered paths and evaluate the union of the top-k paths.

First, CPT achieves a substantially higher hit rate than APP (0.952 vs. 0.699) while retaining more compact paths ($\mathbf{PS}_t$: 0.470 vs. 0.688). Although APP is faster (2,413s vs. 3,114s), this difference may partly reflect that APP does not explicitly account for interactions among nodes within a block, potentially trading broader interaction coverage for lower runtime.

Second, under the same runtime and nearly identical sparsity, CPT substantially outperforms PT: CPT achieves a hit rate of 0.952, whereas PT (Top-61) achieves only 0.012 ($\mathbf{PS}_t$: 0.470 vs. 0.471). As more top-ranked paths are accumulated, PT begins to surpass the edge-level baselines at around Top-150 and eventually reaches a hit rate of 0.940 when all discovered paths are included. However, this requires retaining nearly all path nodes ($\mathbf{PS}_t = 0.987$), yet still yields a lower hit rate than CPT (0.940 vs. 0.952). Without extensively unioning the ranked PT paths, PT tends to maintain a low hit rate because it can examine only a small fraction of the full search space within the runtime budget.

Taken together, these results suggest that, within the evaluated setting, CPT more effectively narrows the search space and identifies compact paths that are relevant to reproducing the original decision than the evaluated path-level baselines.

Model	Stat.	Overall	B1	B2	B3	B4	B5	B6
GPT2-xs	$s^\star$	5.35	11.45(0.94)	5.75(3.21)	5.18(2.98)	2.78(2.45)	2.57(2.42)	4.35(2.65)
	p	0.0192	0.0003	0.0144	0.0216	0.1457	0.1771	0.0401
Pythia-1b	$s^\star$	3.38	5.52(2.77)	6.45(1.90)	4.29(2.12)	6.58(1.48)	2.90(1.50)	3.46(1.64)
	p	0.0860	0.0170	0.0088	0.0420	0.0080	0.1299	0.0806
Pythia-14m	$s^\star$	3.85	3.73(1.36)	3.45(1.21)	3.17(1.21)	3.14(1.10)	2.84(0.95)	4.28(1.08)
	p	0.0587	0.0647	0.0812	0.1024	0.1055	0.1380	0.0422
Model	Stat.	Overall	B7	B8	B9	B10	B11	B12
Pythia-1b	$s^\star$	3.38	3.68(2.18)	2.16(1.33)	2.35(1.37)	3.18(1.96)	1.65(0.98)	4.26(1.53)
	p	0.0860	0.0675	0.2761	0.2232	0.1021	0.5580	0.0428
Model	Stat.	Overall	B13	B14	B15	B16		
Pythia-1b	$s^\star$	3.38	1.64(0.98)	1.43(0.89)	2.24(1.62)	2.34(1.80)		
	p	0.0860	0.5648	0.8705	0.2506	0.2257		

Table F: Block-level(B#) polynomial behavior in language models. The $s^\star$ row reports mean search depth with standard deviation in parentheses; the p row reports the corresponding empirical probability. Model-specific settings are GPT2-xs: $n = 14$, $p_{\text{poly}} = 0.0001$; Pythia-1b: $n = 10$, $p_{\text{poly}} = 0.0010$; Pythia-14m: $n = 6$, $p_{\text{poly}} = 0.0161$. For all reported blocks, empirical p exceeds p_{poly}.

Model	Stat.	Overall	B1	B2	B3	B4	B5	B6
ViT-tiny	$s^\star$	3.85	6.28(1.80)	5.15(1.89)	5.43(1.97)	4.24(1.94)	3.94(1.93)	3.78(1.72)
	p	0.0590	0.0099	0.0222	0.0181	0.0435	0.0551	0.0622
DeiT-tiny	$s^\star$	2.77	5.52(1.74)	4.22(1.79)	3.37(1.96)	2.90(1.74)	2.37(1.52)	1.80(1.31)
	p	0.1470	0.0170	0.0442	0.0866	0.1306	0.2182	0.4349
Model	Stat.	Overall	B7	B8	B9	B10	B11	B12
ViT-tiny	$s^\star$	3.85	3.26(1.80)	3.47(1.58)	3.30(1.59)	2.72(1.46)	2.33(1.48)	2.26(1.53)
	p	0.0591	0.0954	0.0799	0.0918	0.1543	0.2264	0.2444
DeiT-tiny	$s^\star$	2.77	2.33(1.32)	2.29(1.28)	2.10(1.23)	2.26(1.19)	2.37(1.15)	1.67(1.28)
	p	0.1472	0.2271	0.2377	0.2941	0.2454	0.2185	0.5388

Table G: Block-level(B#) polynomial behavior in vision models. The $s^\star$ row reports mean search depth with standard deviation in parentheses; the p row reports the corresponding empirical probability. Both ViT-tiny and DeiT-tiny have $n = 8$ path nodes per block, with $p_{\text{poly}} = 0.0039$. For all reported blocks, empirical p exceeds p_{poly}.

H.12 Block-level Analysis for the Empirical Polynomial Behavior

As shown in Tables F and G, the empirical probability p exceeds the corresponding polynomial bound p_{poly} for every block across all evaluated language and vision models. This indicates that the polynomial behavior observed in Figure 2 is preserved at the block level.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Yes, stated in the abstract and introduction.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Yes, stated in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: Yes, stated in the methodology and appendix.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Yes, stated in the experiments and the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Yes, stated in the appendix along with a code link.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Yes, stated in the experiments and the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Yes, stated in the experiments and the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Yes, stated in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes, we have identified no concerns.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Yes, stated in the introduction, the conclusion, and the appendix.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: Our work does not introduce or release new models or datasets that require release safeguards. Potential misuse of the proposed analysis method is discussed in Appendix Section B.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We use publicly available pretrained models (e.g., GPT2, Pythia) under their respective licenses. All models and tools used are properly credited and cited in the main paper. No modifications were made to their original distributions, and we adhered to their terms of use.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets`

has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We release our code via the GitHub link in Appendix Section H.1. The code includes instructions for reproducing the main experiments.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This work does not involve crowdsourcing or any research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This work does not involve research with human subjects and does not require IRB or equivalent approval.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: We used an LLM solely for grammar correction.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.