REVEALING PATHWAY INTERACTIONS IN NETWORK COMPUTATIONS WITH COMPLETE PATH CONTRIBUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Understanding how neural networks compute is central to the study of artificial and biological intelligence. Networks pass signals through many layers of interconnected units, so explaining what they compute means tracing how information travels from input to output and how pathways interact. Most existing approaches analyze only pairwise interactions between components such as units, layers or attention heads. Although some uses of these methods combine these measurements ad hoc to form a circuit from input to output, these approaches inherently do not capture the higher-order interactions along full causal chains (paths) from input to output, and therefore cannot address how the exponential number of causal paths interact to construct network computation. This leaves a large set of mechanistic questions unanswerable, for example whether inhibitory pathways veto excitatory ones, and how the two combine to yield network decisions. To address this gap, we introduce Complete Path Contribution, a method to decompose the network output into mathematically complete contributions of individual causal chains to study path-level interactions directly and at scale. This decomposition induces definitions of contributions for any circuit at any granularity level via partial summation, with completeness guaranteed at every level, enabling observations that are not feasible with methods that analyze neurons or synapses in isolation. We further present an efficient algorithm that identifies the k strongest causal paths without examining all of them, which would be exponential in the number of layers. Instead, the runtime is empirically linear in the number of chosen paths. By applying Path Contribution to benchmark image classification networks, we quantify excitatory-inhibitory interactions and show how they collectively construct network decisions at a level of detail previously unmeasurable. Furthermore, applying Path Contribution to an interpretable model of the mouse retina reveals how dynamic receptive fields emerge from excitatory-inhibitory interaction motifs among pathways between different cell types in response to time-varying natural stimuli. Overall, Path Contribution reveals how a network's causal pathways interact at scale to transform input to output.

1 UNDERSTANDING NETWORK COMPUTATIONS: FROM NEURONS TO CIRCUITS

Understanding neural network computations matters for advancing neuroscience and AI Kriegeskorte (2015); Richards et al. (2019), for model distillation Hinton et al. (2015), and for AI safety Bereska & Gavves (2024). Artificial and biological networks both compute in a distributed manner, transforming the input through many stages of nonlinear operations DiCarlo et al. (2012); LeCun et al. (2015), often through the combined effect of excitatory (positively contributing) and inhibitory (negatively contributing) units and pathways (Isaacson & Scanziani, 2011; Maheswaranathan et al., 2023). Understanding a network therefore requires understanding not only its individual units, but also how they coordinate in interacting pathways to transform input to output.

Parallel pathways carrying distinct signals are central to the visual system Nassi & Callaway (2009). In primates, the parvocellular pathway is higher resolution, slower, chromatically sensitive, and less adaptive to contrast, whereas the magnocellular pathway is lower resolution, faster, chromatically insensitive, and more adaptive Derrington & Lennie (1984); Kaplan & Shapley (1986); Solomon et al. (2004). In cortex, the ventral stream is associated with object discrimination and the dorsal

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

stream with estimating object position and trajectory Ungerleider & Mishkin (1982); Goodale & Milner (1992).

Interactions between excitation and inhibition have long been known to underlie visual computation. Edge enhancement in the retina arises not from edge-specific excitation, but because inhibition through parallel pathways cancels excitation in uniform regions away from edges Ratliff & Hartline (1959); Kuffler (1953); Barlow (1953). Similarly, object motion sensitivity, which flags moving objects while rejecting background eye movements, arises because all retinal motion generates excitation, but inhibition vetoes it wherever motion is coincident across the visual field Ölveczky et al. (2003); Baccus et al. (2008). That this computation arises only two synapses after edge enhancement, yet was discovered decades later, shows how challenging it is to connect interacting pathways to computation. As features are computed through serial layers, canceling paths become difficult to find because their net effect is zero. Identifying such mechanisms requires measuring the contributions of individual pathways, including those that cancel, rather than only the net effects of units.

Excitation–inhibition interactions also underlie computations in artificial networks. In InceptionV1, a car detector is excited by wheel features below and window features above, and inhibited by the same features in the opposite positions; selectivity arises from subtracting a parallel pathway rather than from excitation alone Olah et al. (2020). In GPT-2 small, some attention heads copy the correct name to the output while others suppress the token the model is about to predict, apparently calibrating its confidence Wang et al. (2022). Such cancellation can hide components from analysis. In language models, ablating one head often triggers compensation by others, so a component's net effect can be near zero while its direct contribution is large McGrath et al. (2023). Excitation and inhibition are therefore properties of paths, not units.

1.1 Two orthogonal axes of explanation

A full explanation of network computation is composed of two orthogonal axes. The parallel (or disjunctive) axis typically involves first specifying a cut of the model's computational graph (such as neurons or weights in a layer). Then, the explanation offers an interpretable account of "what is happening at this stage of the network", such as features represented or causally important neurons. Next, the second axis extends this explanation across depth, which is the serial (or conjunctive) direction of explanation. Traditionally, one applies the method to many layers separately and observes the evolution of the features or causal information contained, and sometimes pairwise interactions between adjacent layers, but this does not constitute a full explanation because it does not answer how representations or actions compose across many layers of the network to drive the output. This higher-order composition happens in a combinatorial number of ways growing exponentially as depth, which makes it an intractable problem so far.

1.2 The landscape of interpretability tools

The first stage of explanation is understanding how a single unit (neuron or component) computes. A canonical framework from neuroscience characterizes a unit by how the input drives it, its *receptive field* (RF), and how it drives the output, its *projective field* (PF) (Lehky & Sejnowski, 1988). RF methods include spike-triggered averaging (De Boer & Kuyper, 1968), instantaneous receptive fields (IRFs) (Maheswaranathan et al., 2023), most exciting inputs (Walker et al., 2019), and sparse autoencoders (Bricken et al., 2023). PF methods include gradient-based attribution (Simonyan et al., 2014; Ancona et al., 2018).

Other methods measure a unit's total causal effect on the output, such as the raw gradient (Simonyan et al., 2014), SmoothGrad (Smilkov et al., 2017), and Integrated Gradients (Sundararajan et al., 2017) for inputs, and LRP (Bach et al., 2015), DeepLIFT (Shrikumar et al., 2017), and Deep SHAP (Lundberg & Lee, 2017) for hidden neurons. Causal interventions (Vig et al., 2020; Geiger et al., 2021) instead compare the original network with a modified one, so they do not explain a unit's PF in the unaltered network. None of these methods show how the RF and PF compose. A related line of work does express a unit's contribution as a composition of its RF and PF, including activation times gradient (Shrikumar et al., 2017), neuron conductance (Dhamdhere et al., 2018; Shrikumar et al., 2018), its application to retinal models (Tanaka et al., 2019; Maheswaranathan

et al., 2023), and contribution decomposition (CODEC) (Melander et al., 2026), which further analyzes population motifs of contributions within a layer.

However, fully explaining a neuron's RF requires more than relating its input to its activation. We must examine the network in between, since the transformation arises from many interacting parallel pathways. Likewise, explaining its PF requires analyzing how the pathways from the neuron to the output interact. A satisfactory method must therefore quantify each pathway's role in a neuron's computation, with guarantees that pathway-level scores fully account for the neuron's effect, and that neuron-level effects in turn aggregate to fully account for the network output.

Existing circuit-level interpretability methods do not offer a satisfactory solution to this problem. One line of related work here is circuit discovery (Conmy et al., 2023; Syed et al., 2024; Marks et al., 2024; Ameisen et al., 2025) and model pruning (Hatefi et al., 2024; 2025). These methods often identify the most important units first via causal interventions (Meng et al., 2022) or attribution (Kramár et al., 2024), and then compose them into a sparse circuit. These circuits may be more interpretable than the full model, and in some cases tailored to semantic interpretation (Geiger et al., 2021), but they do not quantify the contributions of individual paths. A small literature directly quantifies the importance of paths (Lu et al., 2020; 2021; Du et al., 2023; Goldowsky-Dill et al., 2023; Wang et al., 2022), but these works do not advance a notion of hierarchical completeness for the paths of a neuron and, in turn, the neurons of a network.

1.3 A UNIFIED FRAMEWORK FOR ANALYZING NETWORK COMPUTATIONS AT THE UNIT AND CIRCUIT LEVELS AT SCALE

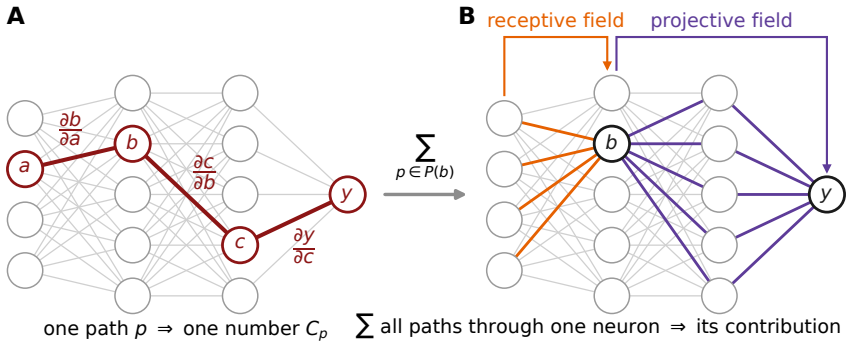

Figure 1: (A) The contribution of a path is the chained gradient along its edges integrated over a straight line path in input space. (B) A neuron's contribution is a composition of its receptive and projective fields. The RF and PF both involve many parallel pathways, and they compose individually into full paths from input to output.

To address this gap, we introduce a principled and unified framework, Complete Path Contribution, to quantify the causal contribution of each neuron, component, subnetwork, or full input-output pathway and analyze them at scale (Figure 1). Our main contributions are as follows:

One algorithm for neurons, components, subnetworks, and paths. Unlike previous methods that first find important components in each layer and then compose them into a circuit ad hoc, we propose a single algorithm that directly measures the contribution of each path, which naturally induces the contribution of any subnetwork as a sum over its paths. In particular, the contribution of an input pixel and a hidden neuron recover Integrated Gradients and neuron conductance (Dhamdhere et al., 2018) respectively, and the contribution of a convolutional channel recovers CODEC (Melander et al., 2026).

Hierarchical completeness. Under our framework, a coarse component's contribution is guaranteed to be an exact sum of the contributions of its finer components (e.g, paths through a neuron, neurons in a convolutional channel), and the sum over any cut of the model's computational graph is guaranteed to equal the output difference with respect to a baseline. See Appendix. A for a the full definition of subnetwork contributions and the proof of hierarchical completeness.

An efficient algorithm for finding the k strongest paths. A full explanation of network computations involves characterizing a number of paths that is exponential in the number of layers. We avoid this intractable problem by devising an efficient algorithm to identify the k strongest paths without exhaustive enumeration. Instead, the runtime is empirically linear in the number of paths requested. Overall, this work enables a unified and consistent view of component-based and circuit-based explanations of network computation, and makes an initial effort towards a full explanation of network computations at the path level via an efficient top-k algorithm.

2 Path Contribution at Scale

2.1 The contribution of a path

The Path Contribution framework starts with defining the contribution of a single path. Consider a feedforward neural network f with input $\boldsymbol{x}$, baseline $\boldsymbol{x}'$, and scalar output $y = f(\boldsymbol{x})$. Let the input be layer 0 and the output be layer L. Let $\mathbf{A}$ be the activations of the network, with $\mathbf{A}^l_{i_l}$ being the activation of neuron i_l in layer l, so $\mathbf{A}^0 = \boldsymbol{x}$ and $\mathbf{A}^L = y$. Let $\mathcal{P}$ be the set of all paths from the input to the output, where a path $p \in \mathcal{P}$ is a sequence of neurons $(i_0, i_1, \ldots, i_L)$ with i_l being a neuron in layer l. Let $\alpha \in [0, 1]$ parameterize a straight-line path from the baseline to the input, so that $\mathbf{A}^0(\alpha) = \boldsymbol{x}' + \alpha(\boldsymbol{x} - \boldsymbol{x}')$.

Definition 1 (Contribution of a path)**.** *By the completeness of Integrated Gradients, the output difference is the sum of the attributions to the input pixels:*

$$\Delta y = \sum_{i_0} \mathrm{IG}^0_{i_0} = \sum_{i_0} (x_{i_0} - x'_{i_0}) \int_0^1 \frac{\partial f\big(\boldsymbol{x}' + \alpha(\boldsymbol{x} - \boldsymbol{x}')\big)}{\partial x_{i_0}} \, d\alpha \tag{1}$$

$$= \sum_{i_0} \int_0^1 \frac{\partial y}{\partial \mathbf{A}^0_{i_0}} \frac{\partial \mathbf{A}^0_{i_0}}{\partial \alpha} \, d\alpha \tag{2}$$

$$= \sum_{i_0} \int_0^1 \sum_{i_1, \ldots, i_L} \frac{\partial \mathbf{A}^L_{i_L}}{\partial \mathbf{A}^{L-1}_{i_{L-1}}} \cdots \frac{\partial \mathbf{A}^1_{i_1}}{\partial \mathbf{A}^0_{i_0}} \frac{\partial \mathbf{A}^0_{i_0}}{\partial \alpha} \, d\alpha \tag{3}$$

$$= \sum_{i_0, i_1, \ldots, i_L} \underbrace{\boxed{\int_0^1 \frac{\partial \mathbf{A}^L_{i_L}}{\partial \mathbf{A}^{L-1}_{i_{L-1}}} \cdots \frac{\partial \mathbf{A}^1_{i_1}}{\partial \mathbf{A}^0_{i_0}} \frac{\partial \mathbf{A}^0_{i_0}}{\partial \alpha} \, d\alpha}}_{\mathbf{C}_{i_0, i_1, \ldots, i_L}} \tag{4}$$

$$= \sum_{i_l} \underbrace{\int_0^1 \frac{\partial y}{\partial \mathbf{A}^l_{i_l}} \frac{\partial \mathbf{A}^l_{i_l}}{\partial \alpha} \, d\alpha}_{\textit{(neuron conductance)}} \tag{5}$$

The boxed quantity is our definition of the contribution of a path $(i_0, i_1, \ldots, i_L)$:

$$\mathbf{C}_{i_0, i_1, \ldots, i_L} \coloneqq \int_0^1 \frac{\partial \mathbf{A}^L_{i_L}}{\partial \mathbf{A}^{L-1}_{i_{L-1}}} \cdots \frac{\partial \mathbf{A}^1_{i_1}}{\partial \mathbf{A}^0_{i_0}} \frac{\partial \mathbf{A}^0_{i_0}}{\partial \alpha} \, d\alpha. \tag{6}$$

We recover Integrated Gradients by summing over all paths that end at the same input pixel, and neuron conductance by summing over all paths that end at the same hidden neuron. Furthermore, path contribution can be taken over a distribution of inputs, for example to measure the average contribution of a path over a dataset. Completeness still holds due to the linearity of expectations.

This definition of path contribution is mathematically equivalent to influence paths (Lu et al., 2020). However, we extend this definition to a hierarchical framework that naturally defines the contribution of any subnetwork, with a guarantee of completeness at every level.

2.2 TOP-k ALGORITHM

The main challenge of fully explaining network computations rests in the intractability of the compositional interactions between components across layers: explaining what happens at every layer of a network does not answer how they interact across layers. At the level of paths, our framework turns this challenge into a single parallel aggregation of an exponential number of paths. Even though characterizing that exponential set remains prohibitive in general, we devise an efficient algorithm to identify the k paths with the highest absolute contribution without exhaustive enumeration.

Without integration and for a single stimulus, the path contribution is purely a product of the Jacobians along the path. Taking the logarithm of the absolute value of the contribution, the problem of finding the top-k paths reduces to finding the k longest paths in a DAG, which can be solved in time polynomial in both k and the network size (Tziavelis et al., 2025). Lu et al. (2020) remarked that an algorithm with such worse case guarantees is impossible due to the summation over the integration steps and inputs, but an algorithm that is efficient in practice is still possible. We present such an algorithm that, within the range we tested, scales up to a 12-layer MLP with 128 neurons per layer ($128^{12} \approx 10^{28}$ paths) in reasonable time on a M3 Pro Macbook Pro.

The algorithm is a best-first branch and bound search, analogous to the A* algorithm, which uses a heuristic bound to find an optimal path without exhaustively searching the graph. The intuition is as follows. Let a *prefix* $(i_0, \ldots, i_l)$ denote the set of all its completions. We now organize the path set as a tree, each node being a prefix. The root is empty, which stands for all paths, and the leaves are individual paths. For every node we compute an upper bound on $|C_p|$ over all its completions. We keep a priority queue ordered by bound, and always expand the most promising node. When a node is a complete path, we put it back in the queue with its exact $|C_p|$ in place of a bound. When a complete path is again at the top of the queue, it is guaranteed to be stronger than any path not yet seen, since its exact contribution outranks every bound still in the queue. The first k paths to be popped from the queue are the top-k paths. The main considerations in making this algorithm efficient are (i) making the bound tight enough that bad subtrees are never expaned, and (ii) keeping the queue small when layers are wide. A complete description of the algorithm and its pseudocode is given in Appendix C.

3 NEURONS AND PATHS: A TOY INTERPRETABILITY CASE STUDY

In Figure 2, we consider a toy model that approximates a continuous exclusive-or (XOR) function and examine what conclusions we would draw from a neuron-based vs. path based approach regarding its computation. With activation analysis, we would be able to characterize the RFs of c and d separately, examining which inputs excite them. Because their RFs are of opposite sign, one may conclude that their RFs differ by an inhibitory connection, but nonetheless cannot deduce that such a connection causally connects the two. Analyzing neuron contributions additionally allows for causal conclusions. Because the activation and contribution of c are of opposite sign, the net causal effect of c on d is negative whenever c is active, which yields a causal statement that c inhibits d as its projective field. However, this analysis misses interactions at the pathway level. In the input region indicated by the green boxes, effects from an inhibitory direct effect from a to d are canceled by an excitatory pathway mediated by c. Analysis of a's contribution would conclude that it is not causally important at that input, even though it is active. This misses the E–I interaction in a's PF that results in a net zero effect. Therefore, a full explanation of network computation necessitates path-level explanations.

4 PATHWAYS INTERACT TO CONSTRUCT THE PROJECTIVE FIELD

A neuron's contribution is composed of its RF and PF. Contributions are often sparser than activations, because an active neuron (RF activated) could have a small downstream gradient (PF not activated) (see (Maheswaranathan et al., 2023) Fig. 6 and (Melander et al., 2026) Fig. 2). In such cases, it is then unknown whether the downstream effect being small is due to individual pathways not crossing threshold, or whether the pathways are individually strong but cancel. We use MNIST-trained MLPs as an illuminating example to investigate this question. Throughout this paper, all MLPs are ReLU networks trained with PyTorch default parameters, using Adam at 10^{-3} learning

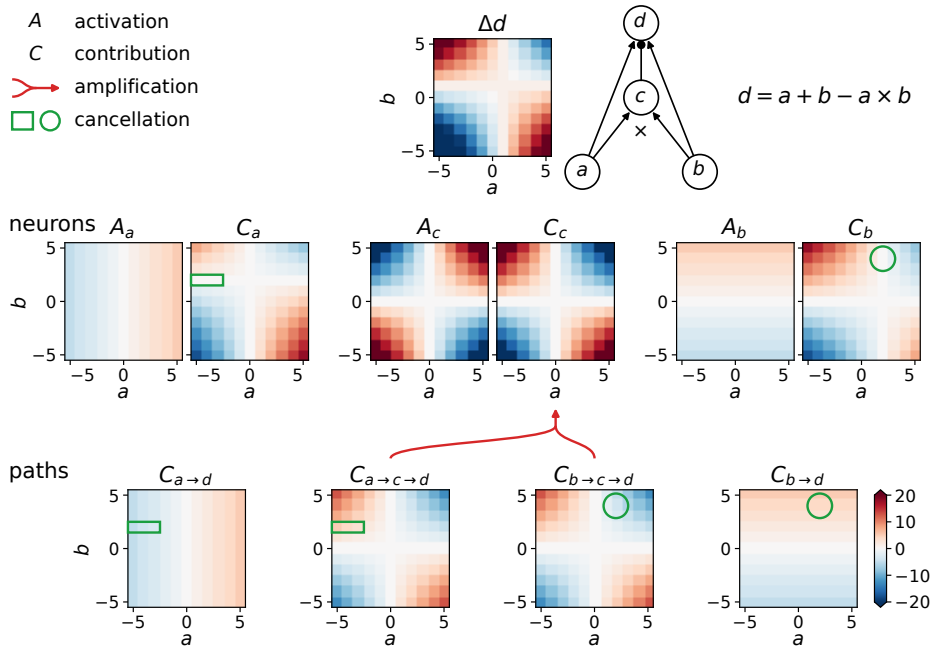

Figure 2: A toy model with two inputs computes $d = a + b - a \times b$. Each heatmap shows the relevant quantity as a function of inputs. Contributions are computed with a zero baseline, so $\Delta d = d$ is the output, shown at the top. The middle row shows the activation and contribution of the three neurons. The bottom row shows the contribution of each of the four paths. Due to hierarchical completeness, $C_{ad} + C_{acd} = C_a$, $C_{acd} + C_{bcd} = C_c$, $C_{bcd} + C_{bd} = C_b$, and $C_a + C_b = \Delta d$. Observe cancellations indicated by the green boxes and green circles, which are not discoverable at the neuron level.

rate, over 5 epochs, at 128 batch size, and reaching over 95% test accuracy. In Figure 3, we study a neuron in the first hidden layer (h1) of a 3-layer MLP that has high activation but low contribution on images of 3 taken to the logit 3 (3s@3). The neuron is selected by a set of criteria: a high level of activity, a substantial contribution to the correct logit, and a low activation-contribution correlation (panel B). The PF of this neuron, h1[13], is revealed by its activations and contributions to all 10 logits across 20 images of each class (panel C). It is roughly linear, showing a consistent pattern over the logits across inputs that is scaled by activation. However, we focus on the nonlinear feature, which is why the neuron is active on 3s@3 but not contributing much as opposed to consistently being inhibitory for other image classes. We then asked why its net contribution is small, are downstream pathways in its PF individually weak and do not cross threshold, or are they strong but cancel? Neuron-level contribution methods cannot answer this question, as they only measure the net effect. Plotting the total E and I effects in its downstream neurons and paths across stimuli and logits (panel E) reveals that the small net contribution results from canceling pathways similar in magnitude to other scenarios where the net effect is large and positive (logit 9) and large and negative (logit 8). In addition, neuron-level net effects (h2 neurons in the plot) are largely single-signed, whereas path contributions have larger E and I totals that cancel to the same residual sum. Therefore, neuron-level contribution analyses overlook the large amounts of E–I interaction at the path level. To visualize this effect, panel F shows the contribution of all paths in h1[13]'s PF, going through a select h2 neuron and h3 neuron. The green box indicates a region where an h2 neuron's total effect (marginal) is small, yet the decomposition into its paths over h3 neurons (a row in the region) shows significant E and I activity that largely cancel. Furthermore, paths offer a sparser description of the PF than neurons. Panel G shows that about 100 times fewer paths are needed by fraction to reconstruct the PF to mean cosine similarity 0.9, whose visualization is shown in panel D. Overall, this analysis demonstrates an example where path contribution produces answers to questions at the pathway level that cannot be answered with neuron-level analyses.

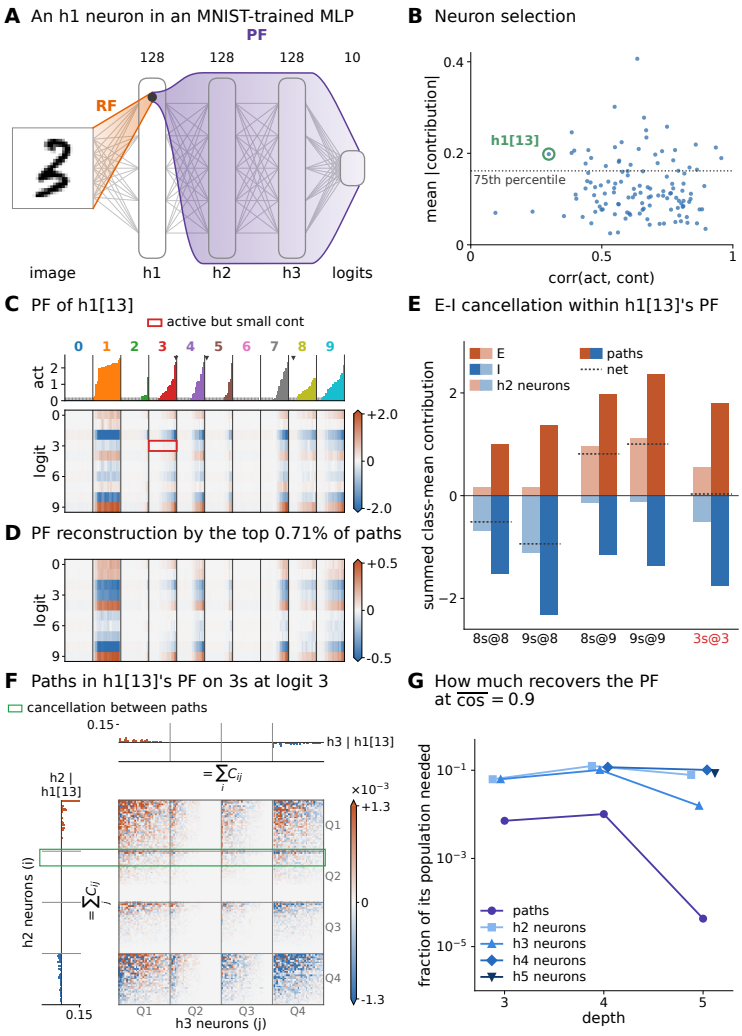

Figure 3: Analysis of pathway interactions in a "silent" neuron's PF in an MNIST-trained MLP. A silent neuron is one that is active but contributes little to the output. (A) The model: A 3-layer MLP with 128 neurons per layer trained on MNIST. (B) The silent neuron selected by a set of criteria: activity—the neuron is active on at least 30% of the validation set, importance—its mean absolute contribution over the validation set to the correct logit is in the top quartile, and silence—a small correlation between its activation and contribution. (C) The PF of the selected neuron h1[13], shown as its contribution on each logit over 20 random images of each class. It has an approximately linear PF: one dominant pattern over the logits, scaled by the activation. However, a striking nonlinear feature is indicated by the red box, where on images of 3s it barely contributes at all despite being consistently inhibitory for other images. (D) A small fraction of paths in the neuron's PF reconstructs the net PF with high accuracy. (E) Net E and I effects in h1[13]'s PF at the neuron (which includes h2 and h3 neurons) and path levels, in five settings. Neuron contributions are mostly decided by the output logit, and are mostly all E or all I. Paths, however, exhibit a high level of cancellation resulting in the same residual sum. On images of 3 taken to the 3 logit (3s@3), E and I effects almost cancel perfectly at both the neuron and path levels. (F) The full path contribution matrix conditioned on h1[13], with paths in a row summing to an h2 neuron's contribution and columns to h3 neurons. Sorted by neuron contribution into quartiles, and within each quartile sorted by the variance. Many h2 and h3 neurons have no net effect, yet path contribution reveals that it is due to large cancellation between paths in some cases (green box) and paths being individually weak in others. An enlarged version of this plot is in Figure 8. (G) A smaller fraction of paths is needed to reconstruct the PF properties of h1[13] compared to downstream neurons, suggesting more sparsity at the path level.

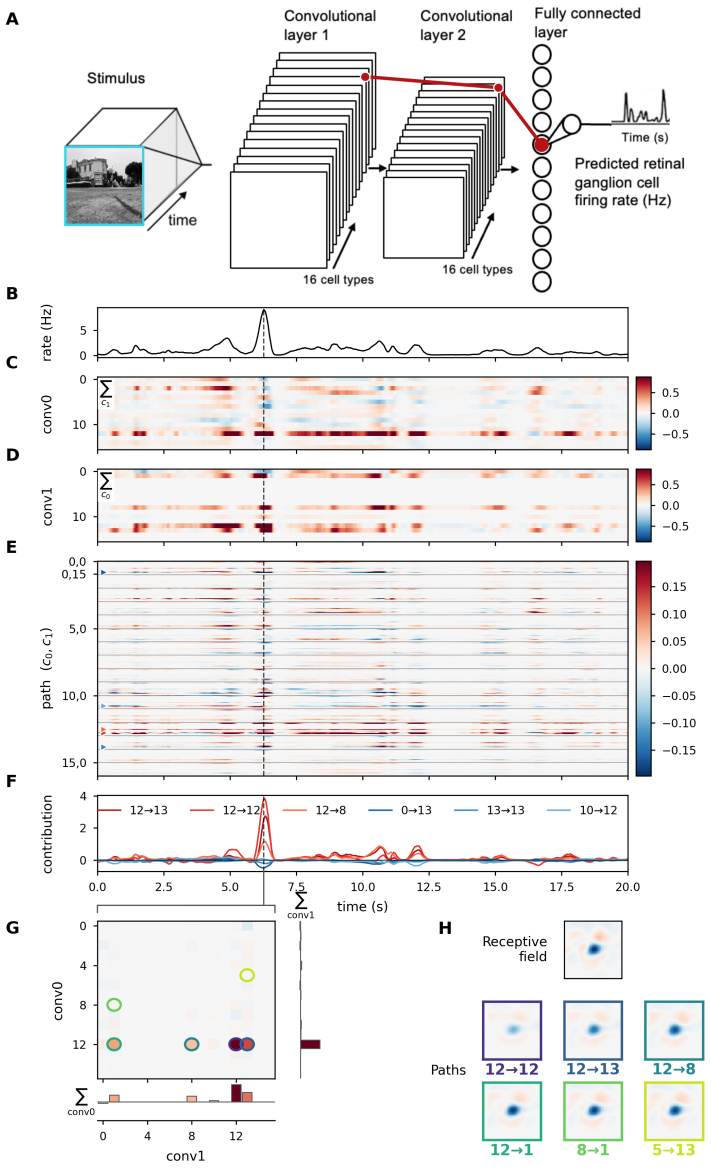

Figure 4: Path contributions to activity and receptive fields of a retinal ganglion cell. (A) Schematic diagram of a 3-layer CNN fit to mouse retinal ganglion cell firing rate responses. A 30-frame (100 x 100) window of a natural movie the movie drives two layers of 16 channels each, read out through a fully connected layer to the rates of all 36 simultaneously recorded cells. Paths are computed for each hidden unit and summed over one pair (c_0, c_1) of channels, one path of the $16 \times 16 = 256$ through the network. The path shown in red, $12 \rightarrow 13$, is the strongest by mean contribution and the first trace in panel F. (B) Predicted firing rate of one cell, $r = 0.57$ against the recorded rate. The dashed line marks the firing peak used in panels G and H. (C, D) Contributions of the 16 conv0 and conv1 units at each time point, obtained by summing the path matrix over the other layer's index. These values are channel-wise contributions for a given layer Melander et al. (2026). (E) All 256 path contributions $P_t[c_0, c_1]$ grouped by conv0 channel; panels C and D are its two marginals. Arrows mark the paths traced in panel F. (F) The three strongest excitatory and inhibitory paths, ranked by mean contribution over the displayed interval. (G) The path matrix at the marked time point, flanked by the two marginals of panels C and D; each marginal is labelled with the index it sums over. Circles mark the paths whose filters are added in panel H. (H) The instantaneous receptive field (IRF) Maheswaranathan et al. (2023) assembled from those paths. Colored borders match paths in panel G. Three paths of 256 sum to r = 0.95 with the IRF.

5 DECOMPOSING CELL-TYPE EFFECTS IN A RETINAL MODEL

Previous work used a generalizable and mechanistically interpretable three-layer CNN model of the retina (Figure 4, panel A) to identify how individual model neurons drive the output by measuring their contributions Maheswaranathan et al. (2023). Although this approach can be applied at any layer, it treats each layer in isolation: it distinguishes the effects of different units within a layer on the retinal output, but does not reveal how units act in series through the network. We therefore used path contribution analysis to examine causal effects from input to output. In a model of mouse retinal ganglion cells with 16 channels (cell types) in each of the two hidden layers, each ganglion cell's response is carried by 256 paths, one for each pair of channels (c_0, c_1) (panel E). Marginalizing the path contributions $P_t[c_0, c_1]$ over c_1 or c_0 recovers the contribution of each conv0 or conv1 cell type, respectively (panels C, D and G), so the paths reveal the effects that combine to produce each cell type's contribution. For a given cell type, the paths passing through it carried both excitation and inhibition, which partially canceled to yield its net contribution (panels E and F). The strongest inhibitory paths converged on the same conv1 channels targeted by the strongest excitatory paths (panel F). Path contributions thus localize where the cancellation occurs, which the cell-type marginals alone cannot reveal. After summing over space, some paths changed sign over time, whereas others remained consistently excitatory or inhibitory (panel E). Inhibitory paths became transiently active at the same moment the excitatory paths peaked, suggesting that inhibition shapes or sharpens the firing event rather than acting as tonic suppression (panel F). Finally, because path contributions are complete, we could ask how many paths are needed to reconstruct the firing rate and the instantaneous receptive field (panels G and H).

We next examined how paths act in concert by applying principal component analysis (PCA) to the $256 \times T$ matrix of path contributions over the clip (Supp. Figure 9). The leading components identified a small number of motifs, or "circuits," each coordinating excitation and inhibition across both hidden layers (panels A and B). For the ganglion cell shown, the first four circuits accounted for 88% of the variance (panel A). Each circuit implemented a distinct receptive field (panel C), and excitation and inhibition played a different role in each (panel E). These interactions are inaccessible to standard contribution analysis, which assigns a single net value to each unit, and they reveal how paths coordinate to shape the model's response.

6 CONCLUSION

We introduce Complete Path Contribution, a unified framework to compute the contribution of any subnetwork, including paths, edges, neurons, and components, with the same algorithm. Furthermore, this framework guarantees hierarchical completeness, where the contributions of a subnetwork's constituents sum to the subnetwork contribution itself, holding across all levels from individual paths to the network output difference. To analyze the exponential number of paths at scale, we devise an efficient algorithm that finds the k most strongest paths in empirically linear time, taking a first step towards a full understanding of network computation. On benchmark image classification models and an interpretable model of the mouse retina, we demonstrate that path contribution offers answers to mechanistic questions at the path level that neuron-level methods cannot answer. We show in an MNIST-trained MLP how cancellation between a neuron's downstream pathways leads to its minimal net contribution despite being activated. We show E–I interaction motifs between attention heads across layers of a ViT that vary and flip sign across inputs. We also show that a single cell type-level path in the retinal model approximates the receptive field of a ganglion cell, and that path-level E–I motifs describe network action that dynamically vary over time. Overall, path contribution reveals important mechanisms of network computation that are invisible to previous methods, and holds promise to enhance our scientific knowledge of how the brain and artificial neural networks compute.

AI USE STATEMENT

In this work, we used generative AI tools to aid or polish writing. We used these tools to review mechanics, improve flow, and to generate sounds bites that facilitate writing. We used generative AI for retrieval and discovery. We used these tools to search literature on a particular topic and summarize them. When important, we read the original source and added it to the references if

applicable. We used these tools for ideation and execution. We used them to generate ideas about the top-k algorithm, which we then manually followed through and verified. We used these tools for execution: we used them to write and refactor code, which we verified by reading it and testing on simple cases. We used these tools to draft sections of the paper. For example, we used them to generate pseudocode for the top-k algorithm that we then manually verified and modified for better presentation We also used these tools for proving mathematical claims, and also to generate figures whose numbers we also verified. We first derived all claims by hand, and then used the tools to generate a streamlined presentation which we verified. We have not used these tools to generate synthetic datasets.

Ethics statement

No human subjects or personally identifiable data are involved, and the image classification experiments use public benchmarks and publicly released checkpoints. The mouse retinal data was collected under a protocol approved by the Stanford University Institutional Animal Care and Use Committee and in accordance with NIH guidelines. The authors declare no competing interests.

Reproducibility statement

We provide precise definitions of path contribution and contribution of subnetworks, how to compute them in practice, along with a detailed explanation and pseudocode for the top-k algorithm. Experiments use MLPs trained with a standard pipeline with a detailed description of hyperparameters, a trained ViT-B available from Torchvision, and an interpretable model of the mouse retina trained on electrophysiology recordings. The implementation, trained retinal and image classification models, the mouse retinal recordings and the analysis pipeline will be released in a public repository on publication.

Acknowledgments

This work was supported by grants from the National Eye Institute (NEI): R01EY022933, R01EY025087 and P30EY026877 (SAB). S.G. thanks the Simons Foundation Collaboration on the Physics of Learning and Neural Computation and a Schmidt Sciences Polymath Award for support. Y.F. was supported by the Stanford HAI-Data Science Scholars Program.

References

Emmanuel Ameisen, Jack Lindsey, Adam Pearce, Wes Gurnee, Nicholas L. Turner, Brian Chen, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Thomas Henighan, Adam Jermyn, Andy Jones, Andrew Persic, Zhenyi Qi, T. Ben Thompson, Sam Zimmerman, Kelley Rivoire, Thomas Conerly, Chris Olah, and Joshua Batson. Circuit tracing: Revealing computational graphs in language models. *Transformer Circuits Thread*, 2025. URL `https://transformer-circuits.pub/2025/attribution-graphs/methods.html`.

Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for Deep Neural Networks. In *International Conference on Learning Representations*, February 2018.

Stephen A. Baccus, Bence P. Ölveczky, Mihai Manu, and Markus Meister. A Retinal Circuit That Computes Object Motion. *Journal of Neuroscience*, 28(27):6807–6817, July 2008. ISSN 0270-6474, 1529-2401. doi: 10.1523/JNEUROSCI.4206-07.2008.

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation. *PLOS ONE*, 10(7):e0130140, July 2015. ISSN 1932-6203. doi: 10.1371/journal.pone.0130140.

Horace B Barlow. Summation and inhibition in the frog's retina. *The Journal of physiology*, 119(1): 69, 1953.

Leonard Bereska and Efstratios Gavves. Mechanistic interpretability for ai safety–a review. *arXiv preprint arXiv:2404.14082*, 2024.

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

Arthur Conmy, Augustine N. Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. Towards Automated Circuit Discovery for Mechanistic Interpretability. In *Thirty-Seventh Conference on Neural Information Processing Systems*, November 2023.

Egbert De Boer and Paul Kuyper. Triggered Correlation. *IEEE Transactions on Biomedical Engineering*, BME-15(3):169–179, July 1968. ISSN 1558-2531. doi: 10.1109/TBME.1968.4502561.

AM Derrington and P Lennie. Spatial and temporal contrast sensitivities of neurones in lateral geniculate nucleus of macaque. *The Journal of physiology*, 357(1):219–240, 1984.

Kedar Dhamdhere, Mukund Sundararajan, and Qiqi Yan. How Important is a Neuron. In *International Conference on Learning Representations*, September 2018.

James J DiCarlo, Davide Zoccolan, and Nicole C Rust. How does the brain solve visual object recognition? *Neuron*, 73(3):415–434, 2012.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*, October 2020.

Kevin Du, Lucas Torroba Hennigen, Niklas Stoehr, Alex Warstadt, and Ryan Cotterell. Generalizing Backpropagation for Gradient-Based Interpretability. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11979–11995, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.669.

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. https://transformer-circuits.pub/2021/framework/index.html.

Atticus Geiger, Hanson Lu, Thomas Icard, and Christopher Potts. Causal Abstractions of Neural Networks, October 2021.

Nicholas Goldowsky-Dill, Chris MacLeod, Lucas Sato, and Aryaman Arora. Localizing Model Behavior with Path Patching, May 2023.

Melvyn A Goodale and A David Milner. Separate visual pathways for perception and action. *Trends in neurosciences*, 15(1):20–25, 1992.

Sayed Mohammad Vakilzadeh Hatefi, Maximilian Dreyer, Reduan Achtibat, Thomas Wiegand, Wojciech Samek, and Sebastian Lapuschkin. Pruning By Explaining Revisited: Optimizing Attribution Methods to Prune CNNs and Transformers, October 2024.

Sayed Mohammad Vakilzadeh Hatefi, Maximilian Dreyer, Reduan Achtibat, Patrick Kahardipraja, Thomas Wiegand, Wojciech Samek, and Sebastian Lapuschkin. Attribution-guided Pruning for Compression, Circuit Discovery, and Targeted Correction in LLMs, June 2025.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Jeffry S Isaacson and Massimo Scanziani. How inhibition shapes cortical activity. *Neuron*, 72(2): 231–243, 2011.

Ehud Kaplan and Robert M Shapley. The primate retina contains two types of ganglion cells, with high and low contrast sensitivity. *Proceedings of the National Academy of Sciences*, 83(8):2755–2757, 1986.

János Kramár, Tom Lieberum, Rohin Shah, and Neel Nanda. AtP*: An efficient and scalable method for localizing LLM behaviour to components, March 2024.

Nikolaus Kriegeskorte. Deep neural networks: a new framework for modeling biological vision and brain information processing. *Annual review of vision science*, 1:417–446, 2015.

Stephen W Kuffler. Discharge patterns and functional organization of mammalian retina. *Journal of neurophysiology*, 16(1):37–68, 1953.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

Sidney R. Lehky and Terrence J. Sejnowski. Network model of shape-from-shading: Neural function arises from both receptive and projective fields. *Nature*, 333(6172):452–454, June 1988. ISSN 1476-4687. doi: 10.1038/333452a0.

Kaiji Lu, Piotr Mardziel, Klas Leino, Matt Fredrikson, and Anupam Datta. Influence Paths for Characterizing Subject-Verb Number Agreement in LSTM Language Models. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4748–4757, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.430.

Kaiji Lu, Zifan Wang, Piotr Mardziel, and Anupam Datta. Influence Patterns for Explaining Information Flow in BERT. In *Advances in Neural Information Processing Systems*, November 2021.

Scott M Lundberg and Su-In Lee. A Unified Approach to Interpreting Model Predictions. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Niru Maheswaranathan, Lane T. McIntosh, Hidenori Tanaka, Satchel Grant, David B. Kastner, Joshua B. Melander, Aran Nayebi, Luke E. Brezovec, Julia H. Wang, Surya Ganguli, and Stephen A. Baccus. Interpreting the retinal neural code for natural scenes: From computations to neurons. *Neuron*, 111(17):2742–2755.e4, September 2023. ISSN 0896-6273. doi: 10.1016/j.neuron.2023.06.007.

Samuel Marks, Can Rager, Eric J. Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. Sparse Feature Circuits: Discovering and Editing Interpretable Causal Graphs in Language Models. In *The Thirteenth International Conference on Learning Representations*, October 2024.

Thomas McGrath, Matthew Rahtz, Janos Kramar, Vladimir Mikulik, and Shane Legg. The hydra effect: Emergent self-repair in language model computations. *arXiv preprint arXiv:2307.15771*, 2023.

Joshua Brendan Melander, Zaki Alaoui, Shenghua Liu, Surya Ganguli, and Stephen Baccus. Causal Interpretation of Neural Network Computations with Contribution Decomposition. In *The Fourteenth International Conference on Learning Representations, ICLR 2026, Rio de Janeiro, Brazil, April 23-27, 2026*, April 2026.

Kevin Meng, David Bau, Alex J. Andonian, and Yonatan Belinkov. Locating and Editing Factual Associations in GPT. In *Advances in Neural Information Processing Systems*, October 2022.

Jonathan J Nassi and Edward M Callaway. Parallel processing strategies of the primate visual system. *Nature reviews neuroscience*, 10(5):360–372, 2009.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom In: An Introduction to Circuits. *Distill*, 5(3):e00024.001, March 2020. ISSN 2476-0757. doi: 10.23915/distill.00024.001.

Bence P. Ölveczky, Stephen A. Baccus, and Markus Meister. Segregation of object and background motion in the retina. *Nature*, 423(6938):401–408, May 2003. ISSN 1476-4687. doi: 10.1038/nature01652.

Floyd Ratliff and H Keffer Hartline. The responses of limulus optic nerve fibers to patterns of illumination on the receptor mosaic. *The Journal of general physiology*, 42(6):1241–1255, 1959.

Blake A Richards, Timothy P Lillicrap, Philippe Beaudoin, Yoshua Bengio, Rafal Bogacz, Amelia Christensen, Claudia Clopath, Rui Ponte Costa, Archy de Berker, Surya Ganguli, et al. A deep learning framework for neuroscience. *Nature neuroscience*, 22(11):1761–1770, 2019.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 3145–3153, Sydney, NSW, Australia, August 2017. JMLR.org.

Avanti Shrikumar, Jocelin Su, and Anshul Kundaje. Computationally Efficient Measures of Internal Neuron Importance, July 2018.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps, April 2014.

Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. SmoothGrad: Removing noise by adding noise, June 2017.

Samuel G Solomon, Jonathan W Peirce, Neel T Dhruv, and Peter Lennie. Profound contrast adaptation early in the visual pathway. *Neuron*, 42(1):155–162, 2004.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 3319–3328, Sydney, NSW, Australia, August 2017. JMLR.org.

Aaquib Syed, Can Rager, and Arthur Conmy. Attribution Patching Outperforms Automated Circuit Discovery. In Yonatan Belinkov, Najoung Kim, Jaap Jumelet, Hosein Mohebbi, Aaron Mueller, and Hanjie Chen (eds.), *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pp. 407–416, Miami, Florida, US, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.blackboxnlp-1.25.

Hidenori Tanaka, Aran Nayebi, Niru Maheswaranathan, Lane McIntosh, Stephen Baccus, and Surya Ganguli. From deep learning to mechanistic understanding in neuroscience: The structure of retinal prediction. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Nikolaos Tziavelis, Wolfgang Gatterbauer, and Mirek Riedewald. Any-k Algorithms for Enumerating Ranked Answers to Conjunctive Queries. *ACM Transactions on Database Systems*, 51(1): 1:1–1:47, September 2025. ISSN 0362-5915. doi: 10.1145/3734517.

Leslie G. Ungerleider and Mortimer Mishkin. Two cortical visual systems. In David J. Ingle, Melvyn A. Goodale, and Richard J. W. Mansfield (eds.), *Analysis of Visual Behavior*, pp. 549–586. MIT Press, Cambridge, MA, 1982.

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. In *Advances in Neural Information Processing Systems*, volume 33, pp. 12388–12401. Curran Associates, Inc., 2020.

Edgar Y. Walker, Fabian H. Sinz, Erick Cobos, Taliah Muhammad, Emmanouil Froudarakis, Paul G. Fahey, Alexander S. Ecker, Jacob Reimer, Xaq Pitkow, and Andreas S. Tolias. Inception loops discover what excites neurons most using deep predictive models. *Nature Neuroscience*, 22(12): 2060–2065, December 2019. ISSN 1546-1726. doi: 10.1038/s41593-019-0517-x.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the Wild: A Circuit for Indirect Object Identification in GPT-2 small, November 2022.

Table 1: Special cases of Definition 2. Layers not listed keep their full sets of neurons and edges.

Subnetwork $\mathcal{H}$	Masks	$\mathsf{C}(\mathcal{H})$
Input pixel i_0	$\boldsymbol{m}^0 = \boldsymbol{e}_{i_0}$	Integrated Gradients
Neuron (l, j)	$\boldsymbol{m}^l = \boldsymbol{e}_j$	Neuron conductance
Component $\mathbb{S} \subseteq \mathcal{V}_l$	$\boldsymbol{m}^l = \mathbf{1}_{\mathbb{S}}$	Component contribution, e.g., conv channel
Edge $(l, j) \to (l+1, k)$	$\boldsymbol{M}^{l,l+1} = \boldsymbol{e}_k \boldsymbol{e}_j^\top$	Edge (weight) contribution
Path $(i_0, \ldots, i_L)$	$\boldsymbol{m}^l = \boldsymbol{e}_{i_l}$ for every l	Path contribution

A THE CONTRIBUTION OF ANY SUBNETWORK

Because the computational graph of a feedforward network is a directed acyclic graph (DAG) $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, any subnetwork (circuit) $\mathcal{H} \subseteq \mathcal{G}$ is also a DAG. In this graph setting, neurons are nodes and weights are edges. Let $(l, i_l) \in \mathcal{V}_l$ denote the i_l-th neuron in layer l, an element of the set of layer-l neurons $\mathcal{V}_l$. Let $(l, i_l) \to (l+1, i_{l+1}) \in \mathcal{E}_{l,l+1}$ denote the edge from neuron (l, i_l) to neuron $(l+1, i_{l+1})$, an element of the set of edges $\mathcal{E}_{l,l+1}$ from layer l to layer $l+1$.

Definition 2 (Contribution of a subnetwork)**.** *A subnetwork $\mathcal{H}$ is given by the allowed neurons and edges in each layer:*

$$\mathcal{H} = \big(\mathcal{V}_0^{\mathcal{H}},\ \mathcal{E}_{0,1}^{\mathcal{H}},\ \mathcal{V}_1^{\mathcal{H}},\ \ldots,\ \mathcal{E}_{L-1,L}^{\mathcal{H}},\ \mathcal{V}_L^{\mathcal{H}}\big), \qquad \mathcal{V}_l^{\mathcal{H}} \subseteq \mathcal{V}_l, \quad \mathcal{E}_{l,l+1}^{\mathcal{H}} \subseteq \mathcal{E}_{l,l+1}. \tag{7}$$

For example, a single neuron is the special case in which every layer but one keeps its full set of neurons and edges. The allowed neurons and edges in each layer induce masks on that layer's neurons and edges, $\boldsymbol{m}^l \in \{0,1\}^{|\mathcal{V}_l|}$ and $\boldsymbol{M}^{l,l+1} \in \{0,1\}^{|\mathcal{V}_{l+1}| \times |\mathcal{V}_l|}$, with

$$m_i^l = \mathbf{1}\big[(l,i) \in \mathcal{V}_l^{\mathcal{H}}\big], \qquad M_{j,i}^{l,l+1} = \mathbf{1}\big[(l,i) \to (l+1,j) \in \mathcal{E}_{l,l+1}^{\mathcal{H}}\big]. \tag{8}$$

Let $\mathcal{P}(\mathcal{H})$ be the set of paths that pass through $\mathcal{H}$, that is, whose every neuron and edge is allowed. The contribution of $\mathcal{H}$ is defined as the sum of the contributions of these paths:

$$\mathsf{C}(\mathcal{H}) = \sum_{p \in \mathcal{P}(\mathcal{H})} \mathsf{C}_p = \sum_{i_0,\ldots,i_L} \mathsf{C}_{i_0,\ldots,i_L} \prod_{l=0}^{L} m_{i_l}^l \prod_{l=0}^{L-1} M_{i_{l+1},i_l}^{l,l+1}. \tag{9}$$

In practice, with N_α integration steps, we compute the contribution of a path with N_α backward passes, one per step. The contribution of any subnetwork also takes only N_α backward passes: at each step, we put the masks where they belong in the backward pass, multiplying the gradient at layer l by $\boldsymbol{m}^l$ and the weights between layers l and $l+1$ by $\boldsymbol{M}^{l,l+1}$. If the Jacobians between all pairs of constrained layers are saved, this process can be parallelized, as in CODEC (Melander et al., 2026).

Theorem 1 (Hierarchical completeness)**.** *Let $\mathcal{H}$ be a subnetwork, and let $\mathcal{H}_1, \ldots, \mathcal{H}_K$ be subnetworks whose path sets partition that of $\mathcal{H}$, $\mathcal{P}(\mathcal{H}) = \bigsqcup_{k=1}^K \mathcal{P}(\mathcal{H}_k)$. Then*

$$\mathsf{C}(\mathcal{H}) = \sum_{k=1}^{K} \mathsf{C}(\mathcal{H}_k). \tag{10}$$

In particular, $\mathsf{C}(\mathcal{G}) = \Delta y$, so the contributions of any family of subnetworks whose path sets partition $\mathcal{P}$ sum to Δy.

Proof. By Definition 2, $\mathsf{C}(\mathcal{H})$ is the sum of C_p over $p \in \mathcal{P}(\mathcal{H})$. Since the $\mathcal{P}(\mathcal{H}_k)$ partition $\mathcal{P}(\mathcal{H})$,

$$\mathsf{C}(\mathcal{H}) = \sum_{p \in \mathcal{P}(\mathcal{H})} \mathsf{C}_p = \sum_{k=1}^{K} \sum_{p \in \mathcal{P}(\mathcal{H}_k)} \mathsf{C}_p = \sum_{k=1}^{K} \mathsf{C}(\mathcal{H}_k).$$

For $\mathcal{H} = \mathcal{G}$, every mask is all ones, and $\mathsf{C}(\mathcal{G}) = \sum_{i_0,\ldots,i_L} \mathsf{C}_{i_0,\ldots,i_L} = \Delta y$ by Equation 4. □

For example, write $C^{0,\dots,l}_{i_0,\dots,i_l}$ for the contribution of the subnetwork that fixes neuron i_0 in layer 0, …, neuron i_l in layer l, and keeps every other layer full. Applying Theorem 1 at each level gives

$$\Delta y = \sum_{i_0} C^0_{i_0} = \sum_{i_0}\sum_{i_1} C^{0,1}_{i_0,i_1} = \sum_{i_0}\sum_{i_1}\sum_{i_2} C^{0,1,2}_{i_0,i_1,i_2} = \cdots = \sum_{i_0,\dots,i_L} C_{i_0,\dots,i_L}. \tag{11}$$

The output difference splits into the input pixels, as in Integrated Gradients; each pixel i_0 splits into its outgoing edges into the first hidden layer; each pixel–neuron pair (i_0, i_1) splits into the three-neuron paths (i_0, i_1, i_2) that extend it; and so on down to full paths.

B Head-Head E–I Interaction Motifs in ViT-B

Transformers have become a dominant architecture in deep learning. Attention heads are a key component of the transformer architecture, and extensive work has been done to uncover the role of attention heads in a transformer's computation (Elhage et al., 2021; Wang et al., 2022). Path Contribution offers a general framework to analyze how head-head interactions contribute at any order (for example, pairwise, triplet, etc.) with guaranteed completeness. As an example, we analyze in Figure 5 how pairwise interactions of attention heads contribute to the classification in a ViT-B/16 model trained on ImageNet-1k (Dosovitskiy et al., 2020) (from Torchvision). We find that even though the overall magnitude of head-head contributions are small compared to their coupling with the residual stream, their contributions naturally decompose into action motifs containing both excitatory and inhibitory paths that dynamically change magnitude and flip sign over different images of the same class. While we do not claim semantic interpretation of these motifs, their existence suggests the ubiquitous utilization of coordinated E–I interactions in network computation.

C Details of the top-k algorithm

The goal of the top-k algorithm is to return the k paths with the largest $|C_p|$ efficiently. To compute the path contribution in practice, we discretize the integral in equation 6 with N_α steps $\alpha_1, \dots, \alpha_{N_\alpha}$ and weights w_t. For example, with the right Riemann sum, $w_t = 1/N_\alpha$ and $\alpha_t = t/N_\alpha$. Writing $J^l_{i_l,i_{l-1}}(t) = \partial A^l_{i_l} / \partial A^{l-1}_{i_{l-1}}$ evaluated at α_t, the contribution of a path $p = (i_0, \dots, i_L)$ is

$$C_p = \sum_{t=1}^{N_\alpha} w_t \,(x_{i_0} - x'_{i_0}) \prod_{l=1}^{L} J^l_{i_l,i_{l-1}}(t). \tag{12}$$

If we are taking the average contribution over a dataset, we can simply treat the batch dimension as more integration steps (perhaps with uneven weights), and the same algorithm applies.

We build a tree that partitions the path set. Each node is a prefix $(i_0, \dots, i_l)$, which stands for the set of all paths that start with $(i_0, \dots, i_l)$. The root is empty, which stands for all paths, and the leaves are individual paths. When we traverse the tree, at each node we compute a product for each integration step (its state):

$$v_t = w_t \,(x_{i_0} - x'_{i_0}) \prod_{l'=1}^{l} J^{l'}_{i_{l'},i_{l'-1}}(t), \qquad t = 1, \dots, N_\alpha. \tag{13}$$

Each *suffix* $(i_{l+1}, \dots, i_L)$ also has a partial product $s_t = \prod_{l'>l} J^{l'}_{i_{l'},i_{l'-1}}(t)$, and when we extend a prefix by a suffix to obtain a full path, we get $C_p = \sum_t v_t s_t$. Therefore, if I am at a prefix, I can bound the contribution of any path that extends it if I have a bound on s_t for every step t.

The bound. A crucial consideration in this algorithm is to come up with a tight bound. If the bound is loose, we will waste time expanding prefixes that do not lead to any of the top-k paths. Because each integration step has its own product, a single bound over all steps is too loose, since there may only be one step where s_t is large yet the bound thinks every step could be just as large. We therefore compute a bound for each step separately, and furthermore used a signed interval for an even tighter bound. For each step t, we precompute once, backward from the output, an interval $[\mathrm{lo}^l_i(t), \mathrm{hi}^l_i(t)]$

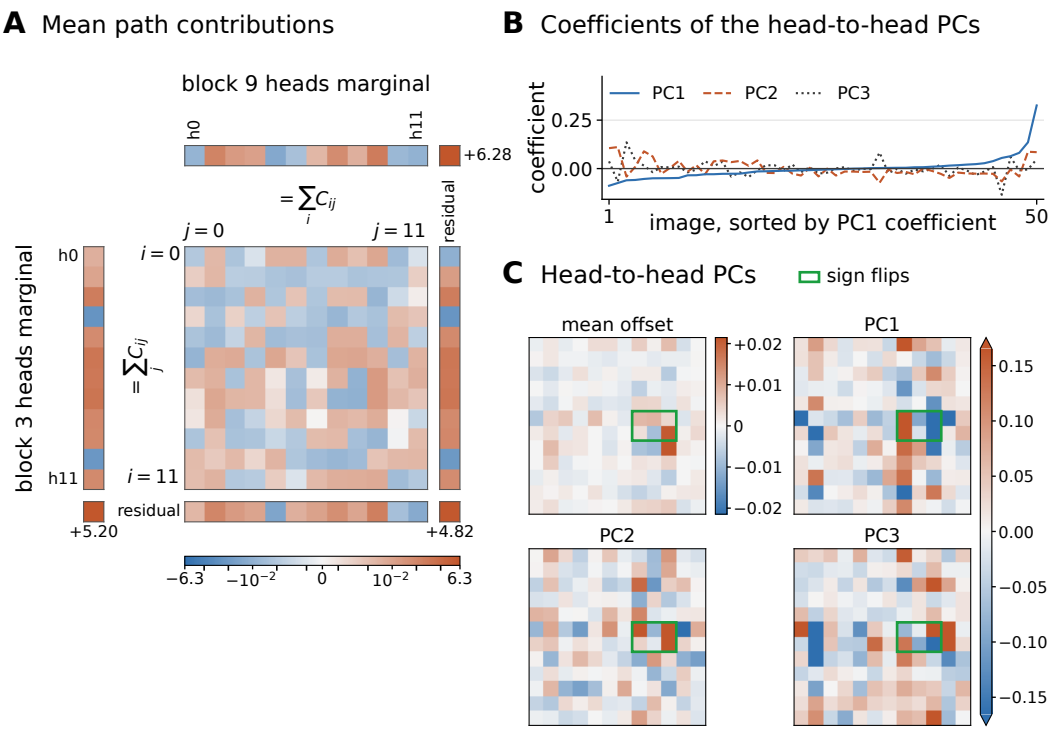

Figure 5: Head-head interaction motifs in ViT-B trained on ImageNet. (A) Mean contributions of paths that pass through specified attention heads at block 3 and block 9, taken over the 50 validation images of the "pill bottle" class. Paths that go through the residual stream in one layer and a head in the other dominate contribution. However, head-head contributions still reveal E–I interaction not visible at a single layer (rows and columns are sums over E and I paths contained in them). (B) PCA coefficients on the head-head contributions over 50 images sorted by PC1. Both PC2 and PC3's coefficients vary across inputs, and sometimes flip sign. (C) The mean offset and top three PCs of head-head contributions.

that contains every s_t reachable from neuron (l, i). With $J^+ = \max(J, 0)$ and $J^- = \min(J, 0)$,

$$\mathrm{hi}_i^l(t) = \max_{j \in \mathcal{V}_{l+1}} \left(J_{j,i}^{l+1,+}(t)\, \mathrm{hi}_j^{l+1}(t) + J_{j,i}^{l+1,-}(t)\, \mathrm{lo}_j^{l+1}(t) \right), \tag{14}$$

$$\mathrm{lo}_i^l(t) = \min_{j \in \mathcal{V}_{l+1}} \left(J_{j,i}^{l+1,+}(t)\, \mathrm{lo}_j^{l+1}(t) + J_{j,i}^{l+1,-}(t)\, \mathrm{hi}_j^{l+1}(t) \right), \tag{15}$$

starting from $\mathrm{lo}^L = \mathrm{hi}^L = 1$ at the output. With this bound on s_t, we have a bound on the contribution of any path that extends a prefix at (l, i_l),

$$|C_p| \le \sum_{t=1}^{N_\alpha} |v_t| \, \max\left(|\mathrm{hi}_{i_l}^l(t)|, \, |\mathrm{lo}_{i_l}^l(t)| \right), \tag{16}$$

so we key (l, i) by this value in the priority queue. Importantly, when a prefix gets expanded into a complete path, this bound becomes exact, i.e. it becomes the true $|C_p|$ for that path.

Splitting one child at a time. We organize the prefixes in a priority queue, ordered by the bound in Equation 16. The algorithm pops the prefix with the largest bound, expands it into its children, and pushes them back into the queue. Naively, when a prefix is popped, all of its children are pushed into the queue, but this is wasteful when a layer is wide, because most of those children will never be expanded. Instead, as part of the preprocessing, we sort each neuron's children by their largest possible bound over all steps, which is a roughly indication of how promising they are. When a prefix is popped, we push only its best child, together with a *sibling entry* standing for all of its

remaining children, whose bound is the largest among them. When a sibling entry is popped, it is again split into its best child and a sibling entry for the rest. This way, the queue grows by about two entries per expansion instead of by the layer width.

Adaptive refinement over integration steps. For a node with d children, the preprocessing step would need to store a bound for each sibling entry at each step, which is $d \times N_\alpha$ numbers per neuron. This is often too large to store for every neuron in a network. Instead, we group the steps into blocks Q of a partition $\mathcal{Q}$, saving only one number per block, at the cost of a looser bound:

$$B_{\mathcal{Q}} = \sum_{Q \in \mathcal{Q}} \Big(\sum_{t \in Q} |v_t| \Big) \max_{c \geq r,\ t \in Q} \rho_c(t), \tag{17}$$

where $\rho_c(t)$ is the bound radius of child c at step t. If each blocks is only step, we recover the full sibling entry case. We start with just one block, and each time a sibling entry reaches the top of the queue, we alternate between refining (splitting one block into two and and putting the entry back with the lower bound) and branching (splitting off its next child). New sibling entries created from branching inherit the same block partition as their parent. This way, only entries that keep being promising after refinement are refined down to single steps. For example, with four steps, let $|v| = (1, 1, 1, 1)$ and two children with radii $\rho_A = (10, 0, 0, 0)$ and $\rho_B = (0, 0, 0, 10)$. At one block, the bound is 40. The two-block bound is still 40, but the four-block bound is 20, and branching gives each child its bound of 10.

Correctness. Branching splits the remaining path set into a partition, so each path is covered by exactly one entry in the queue. Therefore, each path can only be returned once. Furthermore, the refinement and branching process is monotone: a child entry's bound is never larger than its parent's, and a refined entry's bound is never larger than the one before. Therefore, when a complete path is popped from the queue, it is guaranteed to be stronger than any path not yet seen, since its exact contribution outranks every bound still in the queue. Since top paths get popped in order, we can stop the algorithm at any time and the paths already returned will be the top paths, making this an *anytime* algorithm.

Pseudocode.

1. Compute $\mathrm{lo}^l(t)$ and $\mathrm{hi}^l(t)$ backward from the output for every step t, and sort each neuron's children.
2. Push the root into the priority queue, whose children are the input neurons, with state $v_t = w_t(x - x')$.
3. Repeat until the queue is empty: Pop the entry with the largest key.
   (a) If it is a complete path, output it. If k paths have been output, stop.
   (b) If it is a sibling entry due for refinement and not yet at single steps, refine it, set its key to the new bound, mark it to branch next, and push it back.
   (c) Otherwise, branch: take its next child c, update $v_t \leftarrow v_t\, J_{c,\cdot}(t)$, and push this new prefix back with its new key. If there are still children of larger rank, push a sibling entry for them and mark it for refinement next.

Scaling. Figure 6 measures the time to find the top 10^5 paths through single neurons on MNIST-trained ReLU MLPs as various dimensions of the problem are scaled up. Even though the algorithm does not have a worse-case guarantee, it is efficient in practice, finding the top 10^5 paths through a 12-layer MLP with 128 neurons per layer ($128^{12} \approx 10^{28}$ paths) in under half an hour on a M3 Macbook Pro. In panel A, we see that time to find the top 10^5 paths grows linearly in steps and images. Over width runtime barely grows at all at the scale tested. Over depth runtime grows superlinearly but still remains reasonable, even though tests at larger scales are needed to measure the boundary for which this algorithm remains feasible. In panel B, we see that time to find the top k paths starts sublinear in k and approaches linear as k approaches the scale tested, 10^5. An exhaustive algorithm needs to enumerate all paths, which is intractable for even modestly sized networks ($128^{12} \approx 10^{28}$ paths).

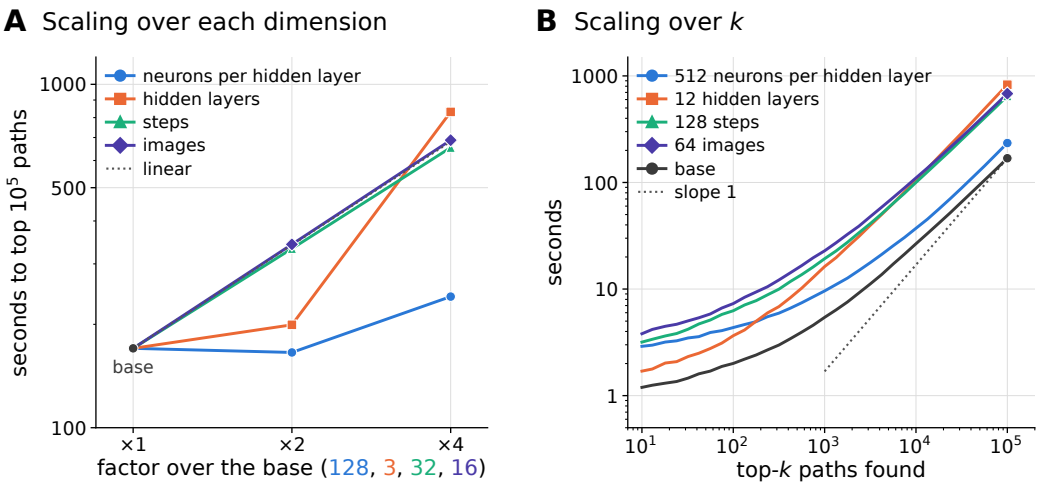

Figure 6: Scaling of the top-k algorithm on MNIST-trained MLPs. All experiments are done on an M3 Pro Macbook Pro. The base run has 3 hidden layers, 128 neurons each, at 32 integration steps, and mean over 16 images. (A) Scaling over each of the four dimensions, with the others held at the base. Time to find the top 10^5 paths grows linearly in steps and images. Over width runtime barely grows at all at the scale tested. Over depth runtime grows superlinearly but still remains reasonable. (B) Scaling over k, the number of top paths to return, for the base run and each dimension maxed out. Runtime starts sublinear in k and approches linear as k approaches the scale tested, 10^5.

D Contribution Concentrates in a Sparse Set of Top-k Paths

Because of completeness, the output change Δy equals sum over all path contributions, and also the sum of neurons in a layer or edges in a layer. To interpret a network, it is often desirable to find a small set of objects that capture important properties of the whole network. Ultimately, the property to explain is the output, so a natural question is how much of Δy concentrates in the top units among the population that distributes that sum. In Figure 7, we study the *coverage* of a set of units, defined as their total contribution as a fraction of Δy, as a function of the number of top units included. For example, for neurons of a give layer, how many top contributing neurons do you need to explain the output difference? We measure coverage as a function of top units included, for the populations of paths, neurons, and edges. We find that in a 3-layer MLP with over a billion paths, only less than 0.1% of top paths are needed to explain half of the output. Around 1% of edges and close to 10% of neurons in a layer are needed for the same coverage, suggesting that network computation is significantly sparser at the path level compared to the edge and neuron levels. In panel B, we address the question of whether a simple beam search over the most important neurons or edges in a layer sequentially yields the top paths. We find that beam search over the top edges needs to enumerate 100 times more paths than the number requested, and beam search over the top neurons 100 times even more to cover all the paths requested. Therefore, our an efficient algorithm to find the top-k paths directly is important for analyzing path contributions at scale.

E Additional Figures

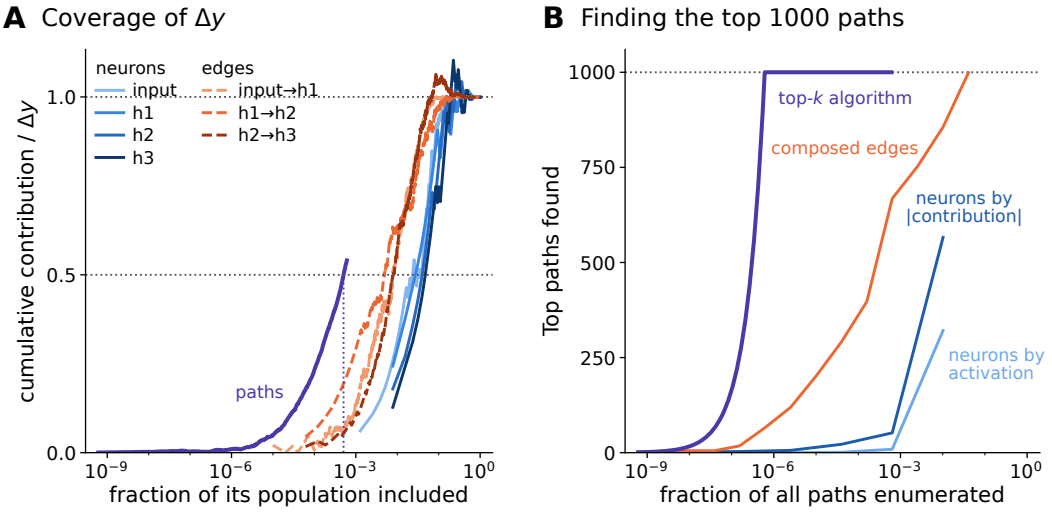

Figure 7: (A) Coverage of Δy as a function of the fraction of a population included: top paths among all paths, top neurons among all neurons in a layer, and top edges among all edges in a layer. About 10 times fewer paths are needed by fraction to cover half of the output compare to edges, and another 10 times fewer than neurons. (B) The number of top paths found using a sequential beam search by most important neurons or edges. A beam search over edges needs to enumerate about 100 times more paths than requested, and neurons 100 times even more.

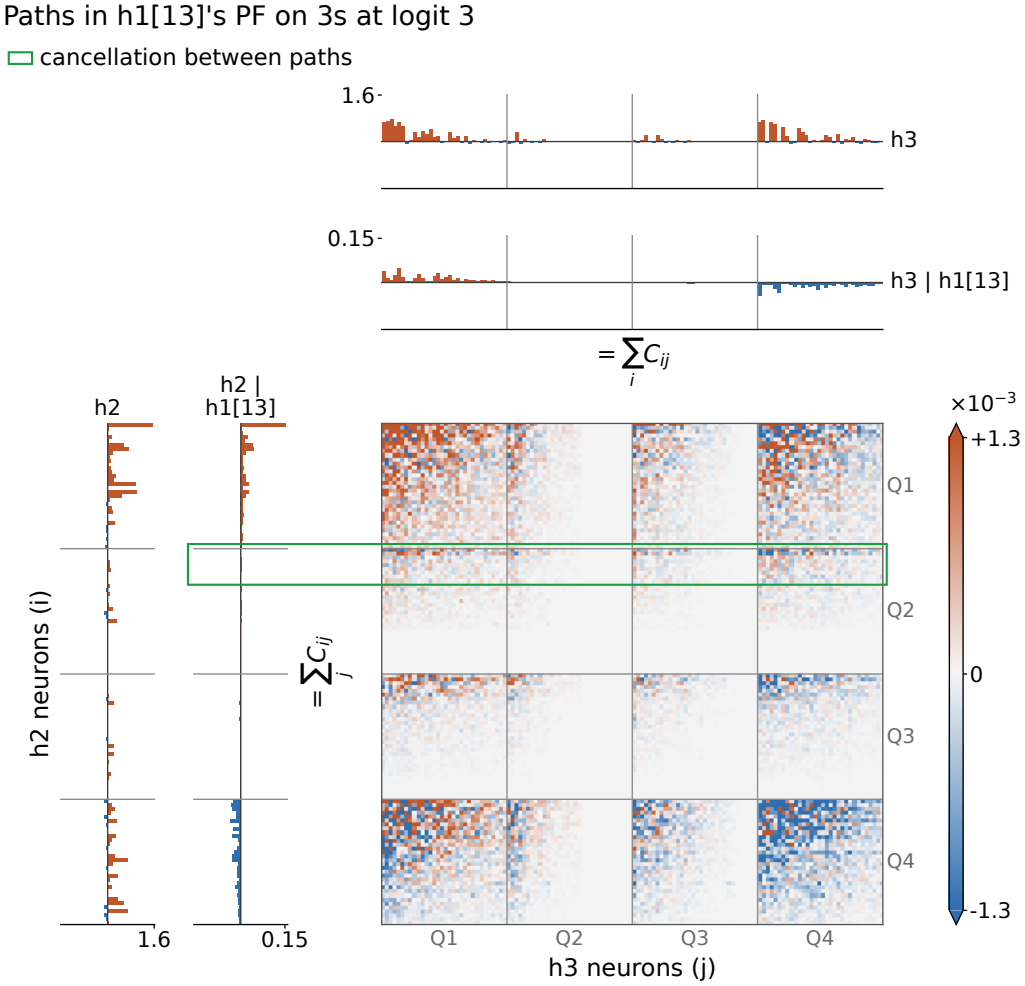

Figure 8: A larger version of Figure 3F.

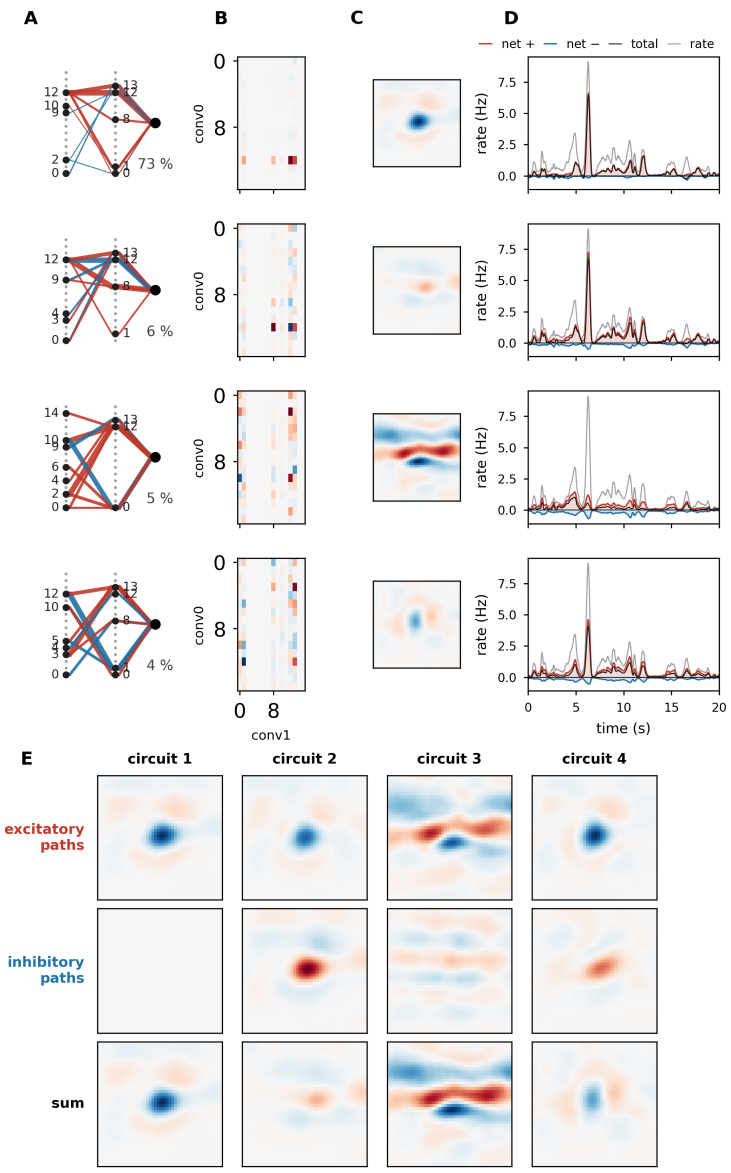

Figure 9: **Circuits formed by paths acting in concert.** Principal components of the $256 \times T$ matrix of path contributions, taken over the whole clip. **(A)** Each circuit drawn over the channel graph, along with the percentage of variance it accounts for. A circuit's paths are those contributing above 5% of its maximum (15, 45, 59 and 70 paths, respectively). One line per path, its width the magnitude of that path's average contribution in the circuit and its color the sign. Line width is the path contribution, not the layer-to-layer weight. **(B)** Path contributions of the same circuits shown as a 16×16 matrix. **(C)** The average receptive field that the circuit implements. **(D)** Contribution of each circuit over time. Shown at each time point are summed contributions for the paths in panel A labelled in red (net positive) or blue (net negative), and the total contribution (black) along with the firing rate (gray). All contributions sum to the firing rate. **(E)** Receptive field contributions for circuit components that are either excitatory or inhibitory, on average, summed over a long stretch of data. Each component has positive and negative values because local increases and decreased in intensity can drive either excitation or inhibition.