Neural Response Interpretation through the Lens of Critical Pathways

Ashkan Khakzar\textsuperscript{1}, Soroosh Baselizadeh\textsuperscript{1}, Saurabh Khanduja\textsuperscript{1}
Christian Rupprecht\textsuperscript{2}, Seong Tae Kim\textsuperscript{1}, Nassir Navab\textsuperscript{1}

\texttt{ashkan.khakzar@tum.de}

\textsuperscript{1}CAMP @ Technical University of Munich
\textsuperscript{2}VGG @ University of Oxford

\section*{Abstract}

Is critical input information encoded in specific sparse pathways within the neural network? In this work, we discuss the problem of identifying these critical pathways and subsequently leverage them for interpreting the network’s response to an input. The pruning objective — selecting the smallest group of neurons for which the response remains equivalent to the original network — has been previously proposed for identifying critical pathways. We demonstrate that sparse pathways derived from pruning do not necessarily encode critical input information. To ensure sparse pathways include critical fragments of the encoded input information, we propose pathway selection via neurons’ contribution to the response. We proceed to explain how critical pathways can reveal critical input features. We prove that pathways selected via neuron contribution are locally linear (in an $\ell_2$-ball), a property that we use for proposing a feature attribution method: “pathway gradient”. We validate our interpretation method using mainstream evaluation experiments. The validation of pathway gradient interpretation method further confirms that selected pathways using neuron contributions correspond to critical input features. The code\textsuperscript{1,2} is publicly available.

\section{Introduction}

Understanding the rationale behind the response of a neural network is of considerable significance. Such transparency is required for adoption and safe deployment in mission-critical domains. Interpreting the response also helps in debugging and designing neural networks, and quenches the intellectual curiosity over how neural networks function \cite{15, 47, 24, 63, 10, 50}.

What insights can we acquire about the underpinnings of a neural network’s response by putting the networks under the microscope and analyzing neurons and pathways? By "pathway", we refer to a union of paths (equivalently a sub-network) that connect the input to the output. Discovering to what patterns neurons correspond — also known as neural decoding in computational neuroscience \cite{6} — has revealed human interpretable concepts encoded in neurons of artificial neural networks, \textit{e.g.} curve and circle detectors \cite{41, 65}. Analyzing the neural pathways has recently revealed human interpretable connections between concepts encoded within each neuron on the pathway, \textit{e.g.} circles being assembled from curves \cite{40}. In this work we discuss pathways responsible for the network’s response given a specific input, but how can we identify these pathways?

Deep rectified neural networks encode the input information using a sparse set of active neurons \cite{12}, and their inference can be deemed as a pursuit algorithm for sparse coding \cite{43, 59}. Such sparse coding of information is akin to how biological neurons encode information in the brain \cite{42, 16}. Yu \textit{et al.} \cite{64} reported that the pathways of active neurons in artificial neural networks overlap significantly for inputs of a given class. Recently, \cite{63} proposed using the pruning objective and knowledge distillation \cite{19} to show that significantly higher levels of sparsity (\textasciitilde87\% for VGG-16 \cite{54} on ImageNet \cite{7}) can be achieved while keeping the prediction intact. These highly sparse pathways are reported as the critical paths and are shown to be different for inputs of different classes and adversarial inputs \cite{45, 63, 64}.

We first investigate, whether these highly sparse pathways derived from the pruning objective indeed encode critical input features. \textit{We show that the pruning objective has solutions that are not critical pathways}, even though they have the same response as the original network. To illustrate \textit{how} the pruning objective can result in such pathways, we construct a pathological greedy pruning algorithm that by design searches for irrelevant pathways while satisfying the pruning objective. Furthermore, we analyze the pathways selected by distillation guided routing \cite{63} and ob-
serve a similar phenomenon. We also use feature visualization \cite{41,30} to decode and semantically analyze the pathways. If these pathways do not encode critical input features, how can we find such pathways? Numerous works have studied the importance of individual neurons for the neural response, and how each neuron encodes information specific to one or a subset of classes \cite{69,4,36,41}. It is therefore intuitive that selected sparse pathways should encompass important/critical neurons for the corresponding response. We thus investigate selecting pathways based on neuron contributions as opposed to the pruning objective. In order to compute the importance of neurons, we use notions of marginal contribution and the Shapley value \cite{51,29,2,60,70}. The first section of the work is devoted to the discussion of critical pathways.

We proceed to answer how critical pathways can help us interpret the response of the network. We prove that in rectified neural networks, pathways selected by neuron contributions are locally linear. We leverage this property and propose an input feature attribution methodology which we refer to as "pathway gradient". We evaluate our attribution methodology with input degradation \cite{48}, sanity checks \cite{1}, and Remove-and-Retrain (ROAR) \cite{20} on Cifar10 \cite{25}, Bridsnap \cite{5}, and ImageNet \cite{7} datasets. By validating our attribution methodology, we also validate that selected pathways using neuron contributions indeed correspond to critical input features. In summary, the main contributions of the paper are:

- We show that the pruning objective does not necessarily extract critical pathways. We illustrate how the pruning can fail by proposing a pathological greedy algorithm that by design searches for irrelevant pathways. Subsequently, we propose selecting pathways based on neuron contributions instead.
- We prove that critical pathways selected by neuron contributions are locally linear ($\ell_2$-ball) in rectified networks. Using local linearity, we propose a feature attribution approach, "pathway gradient", that reveals input features associated with features encoded in the critical pathways.
- We empirically show that computing contribution (approximated Shapley value) of neurons rather than input pixels, improves input feature attribution.

2. Background and related work

Feature visualization / Neural decoding: This task identifies which input patterns activate a neuron. One family of solutions searches for image patches within the dataset that maximize the activations of neurons \cite{67,65,4}. Another series of works generates images that maximize certain neuron activations \cite{38,41,8,53,30}.

Feature attribution: Here, the problem is to find what features in the input are important for the response of a neuron. The notion of importance/contribution is grounded in the effect of removal of a feature on the response. The amount of output change after removing the feature is marginal contribution, and the average of marginal contributions of a feature in all possible coalitions with other features in the input is the Shapley value \cite{51}. Due to computational complexity, several works such as integrated gradients (IntGrad) \cite{61} and DeepSHAP \cite{29} approximate the Shapley value. Recently it has been shown that many approximations break the axioms \cite{60}, leaving integrated gradients as a promising candidate.

A principal class of feature attribution methods use network gradients. \cite{53,3} propose the input gradient itself as attribution. Guided backpropagation (GBP) \cite{57}, LRP \cite{34}, and DeepLIFT \cite{52} modify gradients during back-propagation. Class Activation Maps (CAM) \cite{68} and GradCAM \cite{50} perform a weighted sum of the last convolutional feature maps. Grounded in marginal contribution, other approaches (perturbation methods) mask the input \cite{10,9,44,62} or features \cite{9,49} and observe the output (or information flow \cite{49}).

Evaluation of feature attribution methods: Early evaluation of interpretability relied on human perception, e.g., evaluation by localization accuracy \cite{68} or pointing game \cite{66}. However, the model could be using features outside the human annotation or even non-robust features as in \cite{21}, and such localization-based evaluations penalize the correct attribution method. Moreover, \cite{39,23,55} show/prove that several methods with human interpretable attributions generate the same attribution even after the network’s weights are randomized. These methods are GBP \cite{57}, Deconvolution \cite{65}, DeepTaylor=LRP-α1/β0 \cite{34} and Excitation BackProp \cite{66}. To evaluate such sensitivity to model parameter randomization, sanity checks \cite{1} have been proposed. Recently, input degradation \cite{48} and ROAR \cite{20} experiments have been introduced for evaluating feature importance. Each of these evaluations measure a different perspective which we explain in section 4.2.

3. Selection of critical pathways

3.1. Setup and notation

Consider a neural network $\Phi_{\Theta}(x): \mathbb{R}^D \rightarrow \mathbb{R}$ with ReLU activation functions, parameters $\Theta = \{\theta^1, ..., \theta^L\}$, and $L$ hidden layers with $N_i$ neurons in layer $i \in \{1, ..., L\}$. The total number of neurons is $N = \sum_{i=1}^L N_i$. We use $z^i \in \mathbb{R}^{N_i}$ to represent pre-activation vector at layer $i$, and $a^i \in \mathbb{R}^{N_i}$ for representing the corresponding activation vector, where $a^i = \text{ReLU}(z^i)$, $z^i = \theta^i a^{i-1} + b^i$, and $a^0 = x$. Note, our definition of $\Phi_{\Theta}(x)$ has a single real valued output. The reason is that we are considering the neural pathway to one neuron, and this neuron could in fact be a hidden neuron in a larger network. Thus the response is defined by
$\Phi_\theta(x) = \theta^{L+1}a^L + b^{L+1}$. Each individual neuron in layer $i$ is specified by index $j \in \{1, ..., N_i\}$, thus denoted by $z^i_j$ and $a^i_j$. The vectors $z^i$ and $a^i$ are specifically associated with input $x$. The vector containing activations of all $N$ neurons is denoted by $a = [a^i_j]^N$ (same notation for vectors of other entities related to neurons). $\{a^i_j\}_{j=1}^{N_i}$ denotes the set of neurons in layer $N_i$, and $\{0, 1\}^N$ denotes a set of size $N$ containing 0s and 1s.

3.2. Selection by pruning objective

We first describe the pruning objective and discuss how a solution satisfying this objective does not necessitate it being critical. Let $m \in \{0, 1\}^N$ be a mask that represents the neurons to be kept and pruned. The pruning objective given an input $x$ is then defined as:

$$\arg\min_m \mathcal{L}(\Phi_\theta(x); \Phi_\theta(x; m \odot a)) \quad \text{s.t.} \quad \|m\|_0 \leq \kappa, \quad (1)$$

where $\odot$ and $\mathcal{L}$ denote the Hadamard product and the loss respectively. $\kappa$ controls the sparsity. Equation (1) is a combinatorial optimization problem and a plethora of solutions exist [26, 18, 17, 33, 28].

Does the pruning objective result in sparse pathways that encode the input? Rectified neural networks have sparse positive activations [12, 42, 43, 59], and encode the input in a sparse set of active neurons [42, 12]. Thus, the network represents discriminative and infrequent features (i.e., features with high information) by sparse/in frequent activation values. In this regime, positive activations are detectors of features $\{a^i_j\}$ and encode features that exist in the input. Zero activations represent missingness of features. Therefore, if a neuron is dead (zero), its corresponding features do not exist in the input. We also posit that a dead neuron does not contribute to the output (considering zero activation as a baseline for missingness):

Lemma 1 (Dead Neurons) Considering $a^i$ as the input at layer $i$ to the following layers of the network defined by function $\Phi_\theta^\geq(C) : \mathbb{R}^{N_i} \rightarrow \mathbb{R}$, the Shapley value of a neuron $a^i_j$ defined by $\sum_{C \subseteq \{a^i_j\}} \binom{|C|}{|N_i|} \binom{|N_i|-|C|}{|C|-1} (\Phi_\theta^\geq(C \cup a^i_j) - \Phi_\theta^\geq(C))$ is zero if the neuron is dead ($a^i_j = 0$).

Lemma 1 (proofs for lemmas and propositions are provided in the appendix) shows that dead neurons have a Shapley value of zero, thus do not contribute to the response. We proceed to explain how the pruning objective can select originally dead neurons as critical neurons. Removing an active neuron results in a change in inputs to next layer’s neurons and thus may change their activation value, and this can result in activating an originally dead neuron. We consider a sparse selected pathway undesirable if it contains neurons that were originally dead, but have become active due to the pruning of other neurons. Such a selected pathway is an artificial construct, which does not reflect the original sparse encoding of the input.

Pathological greedy pruning: We construct a greedy algorithm that by design searches for irrelevant pathways while solving the pruning objective (Eq. (1)). The algorithm illustrates how originally dead neurons turn active and become part of the highly sparse selected pathway that produces the same network response. Our greedy approach (Alg. 1) first ranks all neurons based on their relevance score for the response. The relevance is determined by the effect of removing a neuron, approximated by Taylor expansion similar to [33, 37]. The relevance score $s^i_j$ is then:

$$s^i_j = |\Phi_\theta(x) - \Phi_\theta(x; a^i_j \leftarrow 0)| = |a^i_j \nabla_{a^i_j} \Phi_\theta(x)|. \quad (2)$$

Next we remove the neuron(s) with the lowest rank, and alternate between rank computation and removal. However, we tweak the algorithm to find pathways that contain originally dead neurons. At each removal step, we remove the lowest contributing neuron that is not dead (without this crucial step, dead neurons will be pruned before others as their relevance score is zero). By removing a non-zero neuron, the activation pattern can change and some originally

\begin{algorithm}
\caption{Pathological greedy pruning}
\begin{algorithmic}
\State initialize $m^i_j = 1$ \quad $\forall i, j$
\While{$\|m\|_0 \geq \kappa$}
\State $s^i_j \leftarrow |a^i_j \nabla_{a^i_j} \Phi_\theta(x)|$
\If{$s^i_j \leq s^r \land s^i_j \neq 0$} $m^i_j \leftarrow 0$ \; \EndIf
\EndWhile
\end{algorithmic}
\end{algorithm}

Figure 1. Dead Neuron Selection of Pruning Objective. The percentage of originally dead neurons in the selected pathways of different methods reported for sparsity of 90% (see appendix for more sparsity values). Evaluation on pathways extracted from VGG-16 on 1k ImageNet images. All pathways selected by pruning objective contain originally dead (now active) neurons. The observation that when selecting the top 10% of critical neurons, pruning methods select neurons from the dead regions of the network (which after pruning become active) points to the fact that they are selecting pathways unrelated to the input. Our proposal is to use neuron contributions (our NeuronIntGrad, NeuronMCT),
dead neurons can activate and become included in the pathway. Our algorithm illustrates that solving for the pruning objective can result in undesirable pathways.

**Distillation Guided Routing:** DGR [63] relaxes the pruning objective (Eq. 1) by replacing $m = \{0, 1\}^N$ with continuous valued gates $0 \leq \lambda_j^i \in \mathbb{R}$. To induce sparsity, the objective is regularized with an $\ell_2$ norm, i.e. $\|\lambda_j^i\|_1$:

$$\min_{\Lambda} \mathcal{L}(\Phi_\theta(x), \Phi_\theta(x; \Lambda \odot a)) + \gamma \sum_{k=1}^N \|\lambda_j^i\|_1 \text{ s.t. } \lambda_j^i \geq 0,$$

(3)

where $\Lambda = [\lambda_j^i]^N$ is the vector of all $\lambda_j^i$. In our experiments we find that the initial value of $\Lambda$ plays a significant role. Wang et al. [63] use $\lambda_j^i = 1 \forall i, j$ without discussing its role. We denote different initializations with $\text{DGR}(\text{init}=\alpha)$.

**Remark 2** NeuronMCT and NeuronIntGrad assign $c_j^i = 0$ to a neuron with $a_j^i = 0$ (dead neuron).

We denote a pathway by $e = [e_j^i]^N$, where $e_j^i \in \{0, 1\}$ are indicator variables for each neuron indicating whether neuron belongs to pathway $e$. Having computed the contributions $c_j^i$, in order to select a pathway $e = [e_j^i]^N$, with sparsity value $\kappa$, we select neurons with $c_j^i \geq c_\kappa$ where $c_\kappa$ is the contribution value of the corresponding sparsity $\kappa$ in a sorted list of contributions, i.e. if $c_j^i \geq c_\kappa$ then $e_j^i = 1$, else $e_j^i = 0$.

Selecting the values higher than $c_\kappa$ is average-case $\mathcal{O}(n)$. Thus the computational burden of selecting a pathway depends on the contribution assignment procedure. NeuronMCT requires one inference, and for NeuronIntGrad we use 50 inference steps in the experiments. For both methods, the ranking of the $a_j^i$ are performed network-wise, and not layer-wise. We directly compare the contribution of neurons from different layers. This is possible because integrated gradients satisfies completeness, and for each layer $i, \sum_{j=1}^{N_i} c_j^i = \Phi_\theta(x) - \Phi_\theta(x^i \leftarrow 0)$, making the scores directly comparable. For marginal contribution, by definition the contribution is the change in the output, so the contributions are inherently comparable.

**3.4. Pathway selection experiments**

**Pathway analysis** To corroborate the claim that the pruning objective results in undesirable pathways, we evaluate the pathways extracted by the discussed methods from a
Figure 3. **Pathway Decoding.** a) Generating an input that maximizes the network response while restricting the network to a specific pathway selected by different methods. For the original network, the generated input contains class-specific (related to “fig” class) features. When restricted to active neurons (Active Subnet), an input similar to the original input is reconstructed. The reconstructed input for our contribution-based methods (NeuronIntGrad, NeuronMCT) contains only the critical input features (the figs) of the original image. The reconstructed images for pruning-based pathways (except DGR(init=1)) do not show any input related information. b) Feature visualization of the top selected neuron of the final convolutional layer in each pathway (this experiment is only relevant for pathway selection methods). The top neuron in NeuronIntGrad and NeuronMCT pathways encodes features related to the bird’s eye, which is highly relevant to the input image. The top neurons in pruning-based pathways are associated with features not related to the input.

Figure 4. **Feature Attribution via Pathway Gradient.** The gradients of the locally linear critical pathways at different sparsity levels. The pathway is selected using NeuronIntGrad. The pathways are locally linear and their gradient reflects the critical input features. As we select sparser critical pathways, feature attribution reveals input features that are more critical. More examples in the supplementary.

VGG-16 [54] network for 1k ImageNet [7] images. We report results for the pathways of 90% sparsity (more values in appendix). Fig. 1 shows the percentage of previously dead neurons in the selected pathways of all methods. We observe that all pruning based methods converge to selecting undesirable pathways. ~69% of neurons in the top 10% selected neurons of GreedyPruning are originally dead neurons. Another noteworthy observation is the effect of the initial value of gates in the DGR method. When gates are initialized to 1, the pathways do not drift away from the original active pathway as much as they do with random (uniform [0, 1]) initialization (DGR(init=r)). Nevertheless, still ~8% of neurons in the top 10% of DGR(init=1) are originally dead neurons. We analyze the overlap of the selected pathways using the Jaccard similarity between pathway indicators e in Fig. 2a. Note the similarity between NeuronIntGrad and DGR(init=1) compared to DGR(init=r). This suggests that when initializing DGR with 1, the selected pathways do not drift significantly to undesired pathways, and they still roughly contain the critical neurons, explaining why [63] observed meaningful pathways. We also perform a layer-wise similarity analysis between pathways in Fig. 2b. We observe that the overlap between pathways of NeuronIntGrad and NeuronMCT increases as we move towards final layers, implying that the overall difference between their pathways is due to differences in earlier layers.

**Pathway decoding** Feature visualization estimates the input that maximizes a neuron’s response. In order to generate an image \(x_G\) that maximizes the response, the network’s weights are frozen and optimization by gradient descent is done on the input, i.e., \(\arg\max_{x_G} \Phi_\theta(x_G)\). Such optimization without any regularization or priors is prone to generating adversarial artifacts [14, 41]. Hence, we optimize with preconditioning and transformation robustness techniques as in [41] to generated natural looking images. The question we are interested in here is what the pathway corresponding to the input can tell us about that input. This allows us to semantically evaluate the pathways derived from different pathway selection methods. In Fig. 3a, we generate inputs that maximize the network response while the network is restricted to different pathways. When considering the original network, features related to the predicted (“Fig”) class are visualized. When we restrict the network to the active pathway (Active Subnet), optimization attempts to reconstruct the image. At 98% sparsity we observe that, critical features relevant to the predicted class are reconstructed for contribution-based methods. This signifies that the selected
sparse pathway has indeed encoded features relevant to the prediction. However, for pathways selected by the pruning objective (except DGR(init=1)), the reconstructions resemble noise. In Fig. 3b, we perform feature visualization (using entire network) of the selected top neuron in the final convolutional layer of each pathway. We observe that the selected top neuron by NeuronMCT and NeuronIntGrad is semantically highly relevant to the input, as the neuron is responsible for the bird’s eye. The top neuron of DGR(init=1) is also relevant as it relates to feathers. While for the other methods, the top selected neuron is semantically irrelevant, further confirming that the selected pathways are not encoding the input.

4. Interpreting response via critical pathways

In Section 4.1 we show that sparse pathways selected by NeuronIntGrad and NeuronMCT are locally linear. The local linearity is later used in Section 4.2 for input feature attribution via “pathway gradients” and understanding to which features in the input the pathways correspond.

4.1. Local linearity of pathways of critical neurons

Networks with piecewise linear activation functions are piecewise linear in their output domain [35], and thus are linear at a specific point x, and ∀i, j:

\[ Ψ_θ(x) = (\nabla_x Ψ_θ(x))^T x + b_j^L + 1 : z_j^i = (\nabla_x z_j^i)^T x + b_j^i \]  

(5)

Although the network degenerates into a linear function at a given point, it does not mean it is locally linear. Indeed both the value [13] and the gradient [11] are unstable around a point. To discuss the local linearity of the rectified network, we need to define activation pattern [46, 27]:

**Definition 3 (Activation Pattern(AP))** AP is a set of indicators for neurons denoted by Ψ = \{Ω(a_j^i)\} where \( Ω(a_j^i) = 1 \) if \( a_j^i > 0 \) and \( Ω(a_j^i) = 0 \) if \( a_j^i \leq 0 \).

The feasible set S(x) of an AP is the input regions where the AP is constant and thus the function is linear. Let \( B(x,\epsilon) = \{ \tilde{x} \in \mathbb{R}^D : ||\tilde{x} - x||_2 \leq \epsilon \} \) denote the \( \ell_2 \)-ball around x with radius \( \epsilon \), and let \( \tilde{ε}_{x,2} \) denote the largest \( \ell_2 \)-ball around x where the AP is fixed, i.e.

\[ \tilde{ε}_{x,2} = \max_{\epsilon \geq 0 : B(x,\epsilon) \subseteq S(x)} \epsilon \]  

(6)

\( \tilde{ε}_{x,2} \) is the minimum \( \ell_2 \) distance between x and the corresponding hyperplanes of all neurons z_j^i [27]. The hyperplane defined by neuron z_j^i at point x is \( \{ \tilde{x} \in \mathbb{R}^D : (\nabla_x z_j^i)^T \tilde{x} + b_j^i = 0 \} \) or \( \{ \tilde{x} \in \mathbb{R}^D : (\nabla_x z_j^i)^T \tilde{x} + (z_j^i - (\nabla_x z_j^i)^T x) = 0 \} \). If \( \nabla_x z_j^i \neq 0 \) then the distance between x and z_j^i is

\[ ||(\nabla_x z_j^i)^T x + (z_j^i - (\nabla_x z_j^i)^T x)||/||\nabla_x z_j^i||_2 = |z_j^i|/||\nabla_x z_j^i||_2 \]  

(7)

For a neuron z_j^i, if \( \nabla_x z_j^i = 0 \), then \( z_j^i = b_j^i \). In order for the activation of this neuron to change, the \( \nabla_x z_j^i \) and consequently the AP has to change. Therefore the distance is governed by neurons for which \( \nabla_x z_j^i \neq 0 \). [27] prove that \( \tilde{ε}_{x,2} = \min_{i,j} |z_j^i|/||\nabla_x z_j^i||_2 \). Since \( \nabla_x z_j^i \neq 0 \), the existence of a linear region \( \tilde{ε}_{x,2} \) depends on \( |z_j^i| \) not being zero.

**Locally linear network approximation** In order to approximate the original model \( Ψ_θ(x) \) with a selected pathway e, we replace each neuron a_j^i which is not in the pathway, i.e. \( e_j^i = 0 \) with a constant value equal to the initial value (a_j^i) of that neuron. Note the new constant is not a neuron anymore and thus does not propagate gradient. Replacing the neuron with its initial value keeps AP, and neurons z_j^i unchanged. We denote such an approximate model by \( Ψ_θ(x; e) \). Proofs are provided in the appendix.

**Proposition 4** In a ReLU neural network \( Ψ_θ(x) : \mathbb{R}^D \to \mathbb{R} \), for a pathway defined by \( e_j^i \), if \( a_j^i > 0 \) \( \forall e_j^i = 1 \), then there exists a linear region \( \tilde{ε}_{x,2} > 0 \) for \( Ψ_θ(x; e) \) at x.

**Proposition 5** Using NeuronIntGrad and NeuronMCT, if \( c_e > 0 \), then \( Ψ_θ(x; e) \) at x is locally linear.

4.2. Input feature attribution via critical pathways

The gradient of a linear model represents the contributions of each corresponding input feature [32, 3, 53]. Therefore several works perform a first-order Taylor approximation of the network [3, 53]. However, the gradients of networks are unstable and drastically change around an input. To capture the true direction of change, SmoothGrad [56] averages gradients and LIME [32] fits a linear model to the input neighborhood. However, for a locally linear network, the gradients are already stable (constant within a region) in the linear region neighborhood. The gradient reflects the contributions of features in that neighborhood. Based on Proposition 5, the approximate model \( Ψ_θ(x; e) \) is locally linear for NeuronMCT and NeuronIntGrad. Thus, we can derive linear approximations for the model using the critical pathways of the model and use their gradient as attribution maps. We refer to this method as “pathway gradient”. We use different levels of sparsity and observe the most critical input features for the response (Fig. 4). The attribution methodology is visually compared with other attribution methods in Fig. 5. For pathways selected via pruning, as their pathways do not contain critical input information, it would be senseless to use them for feature attribution. There is no guarantee for their linearity as Proposition 4 requires all neurons within the pathway to be active.

**Computing contribution of neurons vs. input pixels:** Computing the Shapley value [51] for pixels does not account for correlations between pixels\(^3\). Ideally, one should

\(^3\)Correlation and interaction are different. The latter is related to effect of features in different coalitions which is accounted for by Shapley value
Figure 5. Comparison with Attribution Methods. Results for our method "pathway gradient" are shown on the right (for pathways selected by NeuronMCT and NeuronIntGrad). Our method provides pixel-level explanations as opposed to GradCAM [50]. GBP [57] is visually pleasing, but it is merely reconstructing image (Sec. 2). Note the improvement of IntGrad (integrated gradients [61]) on the neurons (NeuronIntGrad) over IntGrad on input (InputIntGrad). Also note the improvement of marginal contribution on neurons (NeuronMCT) over direct implementation on input (InputMCT = input $\times$ gradient [52]). More examples for VGG-16 / ResNet-50 in the appendix.

Figure 6. Randomization-Sensitivity Sanity Check. [1] Similarity of attributions before and after network (ResNet-50, ImageNet) parameter randomization. High similarity after randomization suggests that the attribution method is not explaining the network.

know which pixels are correlated (e.g. belong to the same object), and compute a single Shapley value for this group. The Shapley value for a group is known as the Generalized Shapley value [31]. There is an exponential number of groups of pixels. We thus aim to compute the Shapley value only for groups of correlated pixels e.g. an object (more in the appendix). Within the pathway gradient framework, we compute the Shapley value for neurons instead. Neurons inherently correspond to correlated groups of pixels. Thus we are indirectly computing the Shapley value (contribution) of those correlated groups of pixels. In our experiments, using MCT and IntGrad on neurons (denoted by NeuronMCT and NeuronIntGrad) results in considerably better attributions compared to applying them only to input pixels (denoted by InputMCT and InputIntGrad).

**Baseline choice:** The baseline in feature attribution represents the absence of a feature. In the image domain, [65, 61] consider the zero (black image) as baseline. However, zero pixel values do not necessarily reflect the absence of a feature [58, 22]. As explained in Sec. 3.2, zero neurons represent missingness in sparse rectifier networks. The zero baseline is therefore, more justified for neurons than for the input space, and has been also used in [2, 52].

### 4.3. Feature attribution evaluation experiments

Grounding attribution in theoretical notions such as Shapley value is desired [29, 60] (GradCAM, GBP, and Gradients are not based on this notion). However, experiments can point to specific shortcomings in methods, e.g. the approximate Shapley value (IntGrad[61]) for pixels does not perform well in experiments, which may be due to disregarding correlations between input pixels. Each of our experiments examines the methods from a different perspective. As explained in Sec. 2, visual evaluation can be unreliable. Network parameter randomization sanity checks [1] evaluate whether the method is explaining model behavior. Input degradation [48] and Remove-and-Retrain (ROAR) [20] evaluate whether the attribution maps are showing important features in the input (refer to appendix for details). We use TorchRay [9] for implementation of other attribution methods.

**Network parameter randomization sanity checks [1]:** Several attribution methods, such as LRP$-\alpha 1/\beta 0$ [34], Excitation Backprop [66], and (GBP)[57] generate the same result after the network is randomly initialized, thus they are not explaining the network [1, 55]. In this experiment, parameters of the network are successively replaced by random weights, from the last layer to the first layer. At each step, the similarity between the attributions from the original and randomized network are reported (Fig. 6) for 1k ImageNet images (ResNet-50). It is noteworthy that GradCAM seems sensitive here, but the experiment is unfair to it due to the low dimensionality of its maps [1]. Our attributions confidently pass this sanity check.

**Input degradation - LeRF [48]:** Pixels are removed based on their attribution score and the output change is measured. We remove least relevant features first (LeRF). LeRF evaluates methods based on sufficiency of the features for classification and how well methods avoid assigning scores to unimportant features. Results are reported for ResNet-50 on ImageNet, Birdsnap, and Cifar. We observe (Fig. 7(a,b,c)) considerable improvement of NeuronIntGrad
and NeuronMCT over InputIntGrad and InputMCT.

**Remove and retrain (ROAR)** [20]: In input degradation experiments, the change in output might be a result of the network not having seen such artifacts during training. Therefore the ROAR benchmark retrains the network on the modified images. If the accuracy does not drop, the attribution method is not highlighting important features. The experiment is performed for different percentiles of removed pixels. Due to the large number (10^5) of retraining sessions required, we exclude ImageNet and use Cifar10 (ResNet-8) and Bridsnap (ResNet-50) datasets. Gradients, GBP, InputMCT and InputIntGrad are not revealing features that the model learns to use during training. We observe that NeuronIntGrad and NeuronMCT immensely improve their input counterparts (Fig. 7d,e). NeuronIntGrad and NeuronMCT are performing equally to GradCAM. GradCAM benefits from interpolation and has smooth heatmaps, which seem to help with ROAR performance. We see that smoothing (by morphological opening) on our methods (referred to by * in Fig. 7), performs best.

In summary, the attribution experiments show that attribution via critical pathways is a valid methodology with fine-grained attributions. Fine-grained (pixel-level) attributions convey more accurately which features are important to users (Fig. 5). The results support that computing marginal contribution and the Shapley value for neurons improves attribution over directly computing them for input pixels. We posit that (Sec. 4.2) this can be due to the Shapley value for pixels not accounting for correlations between them. Whereas, computing the Shapley value of neurons implicitly considers correlations between pixels. The feature attribution experiments also validate that selected pathways using neuron contributions indeed correspond to critical input features.

**5. Conclusion**

We demonstrate that solving the pruning objective does not necessarily yield pathways that encode critical input features. We propose finding critical pathways based on the neurons contributions to the response, and show how these sparse pathways can be leveraged for interpreting the neural response by proposing the “pathway gradient” method. Our findings on critical pathways and pruning imply that we may need to revisit the reliability of pruned networks, and points to the direction of pruning via attribution. Moreover, considering critical pathways can be of value to other interpretation approaches, e.g. restricting the network to critical pathways can serve as a possible remedy to the vulnerability of perturbation-based attribution methods to adversarial solutions.

**Acknowledgments**

The authors acknowledge the support of the Munich Center for Machine Learning (MCML) and partial support of Siemens Healthineers. C. Rupprecht is supported by Innovate UK (project 71653) on behalf of UK Research and Innovation (UKRI) and by the European Research Council (ERC) IDIU-638009. A. Khakzar and S.T. Kim are corresponding authors.
References


[45] Yuxian Qiu, Jingwen Leng, Cong Guo, Quan Chen, Chao Li, Minyi Guo, and Yuhao Zhu. Adversarial defense through network profiling based path extraction. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2019. 1


13537


[69] Bolei Zhou, Yiyou Sun, David Bau, and Antonio Torralba. Revisiting the Importance of Individual Units in CNNs via Ablation. jun 2018. 2, 4