H-Sets: Hessian-Guided Discovery of Set-Level Feature Interactions in Image Classifiers

Supplementary Material

Appendix

A. Computational cost

Table 10. Time complexity comparison of our method (H-Sets) versus Integrated Gradients (IG) [44], context-aware first-order (CAFO), and second-order explanations(CASO) [38], Archipelago (Arch) [48] and MOXI [43]. Values are mean ± std in seconds over 5 runs.

Dataset	Model	Explanation Methods					
		IG	Arch	CAFO	CASO	MoXI	**H-Sets**
ImageNet	VGG	0.13±0.02	9.44±3.08	0.04±0.01	2.44±0.56	20.51±3.06	23.58±3.34
	ResNet	0.19±0.04	12.18±3.36	0.17±0.08	2.23±0.55	16.35±2.89	16.07±0.93
	DenseNet	0.25±0.05	13.82±4.82	0.08±0.04	3.62±0.82	18.00±3.87	17.15±1.09
	MobileNet	0.09±0.03	7.57±2.39	0.13±0.06	15.97±5.81	9.96±1.68	46.46±5.54
CUB	VGG	0.17±0.04	9.92±4.46	0.03±0.02	2.09±0.68	15.66±3.20	25.00±3.64
	ResNet	0.08±0.02	7.69±3.09	0.06±0.03	1.29±0.29	16.35±2.89	18.48±0.93
	DenseNet	0.27±0.04	13.32±3.74	0.17±0.07	3.78±0.53	15.66±3.20	30.97±8.53
	MobileNet	0.10±0.09	5.92±1.91	0.06±0.04	3.55±1.09	18.21±16.82	17.39±0.99

Tab. 10 summarizes the average runtime per image (in seconds) across 1,000 validation samples. As expected, **H-Sets** requires more computation than first-order methods (e.g., IG) and some interaction-based methods. This overhead, however, is the direct result of computing second-order feature dependencies and enforcing attribution axioms (Sec. 3.3), which together yield more faithful explanations. Unlike region-masking approaches such as Arch and MOXI, H-Sets operates at a fine-grained, pixel-level resolution and produces mathematically interpretable interaction scores rather than heuristic saliency masks.

Despite this, the runtime remains tractable and flexible. Both the number of features per interaction set (ν) and the Hessian threshold (μ) offer effective control over the accuracy–efficiency trade-off: smaller ν or higher μ values reduce runtime substantially with minimal effect on $\text{ROAD}_{\text{AOPC}}$ (see Tab. 3).

B. Qualitative results

Figure 3 show additional qualitative comparisons of attribution methods across diverse ImageNet samples. Each row shows the original image followed by saliency maps produced by different methods.

Figure 3. Qualitative comparison of attribution methods . Each row shows, from left to right: the original image, and saliency maps from Integrated Gradients (IG) [44], Archipelago [48], Context-aware First Order explanations (CAFO) [38], Context-aware Second Order explanations (CASO) [38], MOXI [43] and our proposed method, **H-Sets**. Darker red indicates higher attribution.

C. Sensitivity to Non-Semantic Features

A potential concern regarding the use of Segment Anything (SAM) [24] as a spatial prior is whether it "forces" the explanation to be semantic, thereby masking a classifier's reliance on non-semantic shortcuts or spurious correlations.

To investigate this, we evaluate H-Sets on the DecoyMNIST [9]. In this setup, a digit classifier is intentionally trained on data where a fixed, non-semantic "decoy" patch is correlated with the class label. A faithful attribution method must be able to ignore the semantic digit and localize the spurious patch if that is what the model is truly utilizing for its prediction.

As shown in Figure 4, H-Sets correctly identifies these non-semantic regions as highly salient. This is because the core of our interaction discovery is *curvature-driven* via the input Hessian. While SAM provides a grouping prior to organize these interactions into coherent sets, the actual attribution scores—calculated using Harsanyi dividends and IDG-Vis—are derived strictly from second-order derivatives.

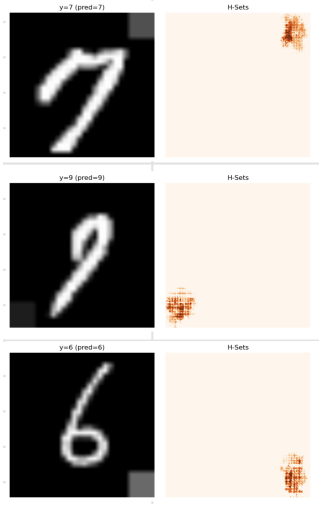


Figure 4. H-Sets attribution on DecoyMNIST. Despite the presence of a strong semantic prior (the digit), H-Sets successfully localizes the spurious decoy patch, demonstrating that the method captures the model's true reliance on non-semantic shortcuts.

D. Algorithms

Algorithms for H-Sets and IDG-Vis are provided in Algorithm 1 and Algorithm 2.

E. Proof for axioms

In this section, we show that our proposed attribution method satisfies all the axioms listed in Sec. 3.3.

Proposition 1. *$a(\mathcal{I})$ satisfies* Axioms *1-4*

Algorithm 1 H-Sets Algorithm

function GETSET(example $\mathbf{x}$, model f, gradient $\nabla f(\mathbf{x})$, index j, threshold μ, max elements ν, interaction set $\mathcal{I}$)
  $\mathbf{H}_j \leftarrow \frac{\partial}{\partial x_i}(\nabla f(\mathbf{x}))_j$ ▷ *Create Hessian*
  $\mathbf{H}'_j \leftarrow \{i \in \{1, 2, \ldots, d\} | \mathbf{H}_j[i] > \mu\}$ ▷ *Consider interactions above threshold*
  next $\leftarrow \{\}$
  for all $n \in \mathbf{H}'_j$ **do**
    if $n = 0$ **then**
      continue
    else if $n \notin \mathcal{I}$ and $|\mathcal{I}| < \nu$ **then**
      $\mathcal{I} \leftarrow \mathcal{I} \cup n$ ▷ *Add interaction to current feature interaction set*
      next $\leftarrow$ next $\cup\, n$
    end if
  end for
  for all $m \in$ next **do**
    GETSET($\mathbf{x}$, f, $\nabla f(\mathbf{x})$, argmin$_j$\{next$[j] = m\}$, μ, ν, $\mathcal{I}$) ▷ *Find more interactions*
  end for
  return $\mathcal{I}$
end function

function GENERATESETS(example $\mathbf{x}$, model f, threshold μ, max elements ν, number of interaction sets k)
  masks $\leftarrow$ SAM($\mathbf{x}$);
  indexes $\leftarrow$ sort(IntegratedGradients($\mathbf{x}$, f)) ▷ *Get Starting Features*
  $\mathcal{S} \leftarrow \{\}$
  for index $\in$ indexes **do**
    if index $\in \{i | i \in m, \forall m \in$ masks$\}$ **then**
      $\mathcal{I} \leftarrow$ GETSET($\mathbf{x}$, f, $\nabla f(\mathbf{x})$, index, μ, ν, $\mathcal{I}$))
      $\mathcal{S} \leftarrow \mathcal{S} \cup \mathcal{I}$ ▷ *Add Interaction Set into Total Set*
      masks $\leftarrow \{m \in$ masks$|$index $\notin m\}$
    end if
    if $|S| = k$ **then**
      break
    end if
  end for
  return $\mathcal{S}$
end function

Algorithm 2 Integrated Directional Gradients for Vision (IDG-Vis)

function ATTRIBUTESETS(example $\mathbf{x}$, model f, interaction set $\mathcal{I}$)
  $\mathbf{x}' \leftarrow \mathbf{0}^{H \times W \times C}$ ▷ *Create baseline*
  $a \leftarrow 0$
  for all $T \sim \mathcal{U}(\mathcal{P}(\mathcal{I}))$ **do**
    $\mathbf{a} \leftarrow x_i \in T \; ? \; (\mathbf{x} - \mathbf{x}') : 0$
    $\hat{\mathbf{a}} \leftarrow \frac{\mathbf{a}}{||\mathbf{a}||}$ ▷ *Create directional vector*
    $A \leftarrow 0$
    for $k \leftarrow 0$ **to** m **do**
      $\nabla_T f(\mathbf{x}) \leftarrow \nabla f(\mathbf{x}' + \frac{k}{m}(\mathbf{x} - \mathbf{x}')) \cdot \hat{\mathbf{a}}$ ▷ *Compute Directional Gradient*
      $A \leftarrow A + \nabla_T f(\mathbf{x})$ ▷ *Create Integrated Directional Gradient for Vision*
    end for
    $A \leftarrow \frac{1}{m+1} \cdot (A)$
    $a \leftarrow a + A$
  end for
end function

Since the TU-game is always positive, it must be true that non-negativity, normality, monotonicity, and superadditivity are satisfied [7].

Lemma 1 (Approximate Completeness)**.** *Let $\mathcal{S}$ be the set of feature interaction sets detected by H-Sets for an input $\mathbf{x}$ and baseline $\mathbf{x}'$. Then the total interaction attribution satisfies:*

$$\sum_{\mathcal{I} \in \mathcal{S}} a(\mathcal{I}) \leq f(\mathbf{x}) - f(\mathbf{x}')$$

with equality if and only if the union of subsets $T \sim \mathcal{U}(\mathcal{P}(\mathcal{I}))$ for all $\mathcal{I} \in \mathcal{S}$ covers the full support of the input difference vector $\mathbf{x} - \mathbf{x}'$.

Proof. We define the linear interpolation path $\mathbf{x}(\alpha) = \mathbf{x}' + \alpha(\mathbf{x} - \mathbf{x}')$, for $\alpha \in [0, 1]$.

By definition, the directional gradient for interaction set $\mathcal{I}$ is:

$$\nabla_{\mathcal{I}} f(\mathbf{x}(\alpha)) = |\nabla f(\mathbf{x}(\alpha)) \cdot \hat{\mathbf{a}}_{\mathcal{I}}|$$

and the integrated directional gradient is:

$$a(\mathcal{I}) \approx \sum_{T \sim \mathcal{U}(\mathcal{P}(\mathcal{I}))} \int_0^1 |\nabla f(\mathbf{x}(\alpha)) \cdot \hat{\mathbf{a}}_T| \, d\alpha$$

Summing over all sets $\mathcal{I} \in \mathcal{S}$ yields ("integral of sums" from "sum of integrals"):

$$\sum_{\mathcal{I} \in \mathcal{S}} a(\mathcal{I}) \approx \int_0^1 \sum_{\mathcal{I} \in \mathcal{S}} \sum_{T \sim \mathcal{U}(\mathcal{P}(\mathcal{I}))} |\nabla f(\mathbf{x}(\alpha)) \cdot \hat{\mathbf{a}}_T| \, d\alpha$$

The total directional gradient, from Integrated Gradients, is given by:

$$\int_0^1 \nabla f(\mathbf{x}(\alpha)) \cdot \frac{d\mathbf{x}(\alpha)}{d\alpha} \, d\alpha = f(\mathbf{x}) - f(\mathbf{x}')$$

Our method calculates the total attribution by summing the scores $a(I)$ for a detected subset of interaction sets S. However, we calculate a sum of attributions for only a select group of interactions. Hence, $a(I)$ under-approximates the total directional gradient, unless the set of direction vectors $\hat{\mathbf{a}}_T$ spans the direction of $(\mathbf{x} - \mathbf{x}')$. Since $\mathcal{S}$ only includes a subset of all possible interaction sets, we obtain:

$$\sum_{\mathcal{I} \in \mathcal{S}} a(\mathcal{I}) \leq f(\mathbf{x}) - f(\mathbf{x}')$$

Equality holds when the sampled subset directions fully represent the input difference vector. □

Lemma 2. *If an input $\mathbf{x}$ and a baseline $\mathbf{x}'$ are equal everywhere except $\mathbf{x}_{\mathcal{I}} \neq \mathbf{x}'_{\mathcal{I}}$ and if $f(\mathbf{x}) \neq f(\mathbf{x}')$, then $a(\mathcal{I}) \neq 0$. In other words, $a(\mathcal{I})$ satisfies Axiom 6.*

Proof. We have two inputs, $\mathbf{x}$ and $\mathbf{x}'$, such that $\mathbf{x}$ and $\mathbf{x}'$ differ only on the subset of features $\mathcal{I}$: $\mathbf{x}_{\mathcal{I}} \neq \mathbf{x}'_{\mathcal{I}}$, while for any $j \notin \mathcal{I}$, $\mathbf{x}_j = \mathbf{x}'_j$. Additionally, it is given that the model's outputs at these points are different: $f(\mathbf{x}) \neq f(\mathbf{x}')$.

The attribution $a(\mathcal{I})$ for a set of features $\mathcal{I}$ is defined by the Integrated Directional Gradients (IDG) approach:

$$a(\mathcal{I}) = \int_{\alpha=0}^1 \nabla_{\mathcal{I}} f(\mathbf{x}' + \alpha(\mathbf{x} - \mathbf{x}')) \, d\alpha$$

This integral measures the cumulative effect of changing the features in $\mathcal{I}$ from $\mathbf{x}'$ to $\mathbf{x}$ on the model's output f, along a linear path parameterized by α from 0 to 1. Since $f(\mathbf{x}) \neq f(\mathbf{x}')$, we know that there is a non-zero change in the model's output as we move along the path from $\mathbf{x}'$ to $\mathbf{x}$. Because $\mathbf{x}$ and $\mathbf{x}'$ differ only on the features in $\mathcal{I}$, any change in f along this path is attributed solely to the variations in the features within $\mathcal{I}$. Hence, this integral must capture a non-zero contribution from $\mathcal{I}$, resulting in $a(\mathcal{I}) \neq 0$. □

Lemma 3. *Two neural networks $f(\cdot)$ and $f'(\cdot)$, with corresponding value functions a' and a'', are functionally equivalent if $f'(\mathbf{x}) = f''(\mathbf{x})$ for all $\mathbf{x}$. Then, $a'(\mathcal{I}) = a''(\mathcal{I})$ for sets $\mathcal{I} \in \mathcal{S}$. $a(\mathcal{I})$ satisfies Axiom 7.*

Proof. If two neural networks f' and f'' are functionally equivalent, meaning $f'(\mathbf{x}) = f''(\mathbf{x})$ for all inputs $\mathbf{x}$, their attributions for any feature set $\mathcal{I} \in \mathcal{S}$ will also be identical, provided they use the same path for gradient computation. IDG computes attribution by integrating over a defined path from a baseline $\mathbf{x}'$ to the input $\mathbf{x}$, so having both networks follow this same path ensures that their gradients are identical at each point along it. Consequently, the attributions $a'(\mathcal{I})$ and $a''(\mathcal{I})$ will match for all feature sets $\mathcal{I} \in \mathcal{S}$, since the path-integrated contributions are equal. Thus, $a'(\mathcal{I}) = a''(\mathcal{I})$ for all $\mathcal{I} \in \mathcal{S}$. □

Lemma 4. *Given two neural networks $f'(\cdot)$ and $f''(\cdot)$, and $f(\mathbf{x}) = c \cdot f'(\mathbf{x}) + d \cdot f''(\mathbf{x})$, then $a(\mathcal{I}) = c \cdot a'(\mathcal{I}) + d \cdot a''(\mathcal{I})$. $a(\mathcal{I})$ satisfies Axiom 8.*

Proof. Attribution $a(\mathcal{I})$ for a feature set $\mathcal{I}$ is computed using the IDG, which involves integrating the gradient along a path from a baseline $\mathbf{x}'$ to the input $\mathbf{x}$:

$$a(\mathcal{I}) = \int_{\alpha=0}^1 \nabla_{\mathcal{I}} f(\mathbf{x}' + \alpha(\mathbf{x} - \mathbf{x}')) \, d\alpha.$$

Similarly, for f' and f'', we have:

$$a'(\mathcal{I}) = \int_{\alpha=0}^1 \nabla_{\mathcal{I}} f'(\mathbf{x}' + \alpha(\mathbf{x} - \mathbf{x}')) \, d\alpha$$

and

$$a''(\mathcal{I}) = \int_{\alpha=0}^1 \nabla_{\mathcal{I}} f''(\mathbf{x}' + \alpha(\mathbf{x} - \mathbf{x}')) \, d\alpha.$$

Since $f(\mathbf{x}) = c \cdot f'(\mathbf{x}) + d \cdot f''(\mathbf{x})$, the gradient of f with respect to the features in $\mathcal{I}$ is:

$$\nabla_{\mathcal{I}} f(\mathbf{x}) = c \cdot \nabla_{\mathcal{I}} f'(\mathbf{x}) + d \cdot \nabla_{\mathcal{I}} f''(\mathbf{x}).$$

This linearity holds for any point along the path $\mathbf{x}(\alpha) = \mathbf{x}' + \alpha(\mathbf{x} - \mathbf{x}')$, so we have:

$$\nabla_{\mathcal{I}} f(\mathbf{x}(\alpha)) = c \cdot \nabla_{\mathcal{I}} f'(\mathbf{x}(\alpha)) + d \cdot \nabla_{\mathcal{I}} f''(\mathbf{x}(\alpha)).$$

We can now substitute this expression into the integral that defines $a(\mathcal{I})$:

$$a(\mathcal{I}) = \int_{\alpha=0}^{1} \nabla_{\mathcal{I}} f(\mathbf{x}(\alpha))\, d\alpha$$

$$= \int_{\alpha=0}^{1} \left(c \cdot \nabla_{\mathcal{I}} f'(\mathbf{x}(\alpha)) + d \cdot \nabla_{\mathcal{I}} f''(\mathbf{x}(\alpha))\right)\, d\alpha.$$

By the linearity of integration, we can separate the terms inside the integral:

$$a(\mathcal{I}) = c \cdot \int_{\alpha=0}^{1} \nabla_{\mathcal{I}} f'(\mathbf{x}(\alpha))\, d\alpha + d \cdot \int_{\alpha=0}^{1} \nabla_{\mathcal{I}} f''(\mathbf{x}(\alpha))\, d\alpha.$$

This simplifies to:

$$a(\mathcal{I}) = c \cdot a'(\mathcal{I}) + d \cdot a''(\mathcal{I}).$$

□

Lemma 5. *Given two symmetric feature interaction sets $\mathcal{I}_1$ and $\mathcal{I}_2$, the value function for neural network f will be $a(\mathcal{I}_1) = a(\mathcal{I}_2)$. In other words, $a(\mathcal{I})$ satisfies Axiom 9.*

Proof. Since $\mathcal{I}_1$ and $\mathcal{I}_2$ are symmetric, swapping the features in $\mathcal{I}_1$ with those in $\mathcal{I}_2$ does not change the output of f. This symmetry implies that the gradients with respect to the features in $\mathcal{I}_1$ and $\mathcal{I}_2$ are equal:

$$\nabla_{\mathcal{I}_1} f(\mathbf{x}) = \nabla_{\mathcal{I}_2} f(\mathbf{x}).$$

Furthermore, because there must be a subset $T_1 \subseteq \mathcal{I}_1$ that has a corresponding symmetric subset $T_2 \subseteq \mathcal{I}_2$ (with $|T_1| = |T_2|$), then we also have:

$$\nabla_{T_1} f(\mathbf{x}) = \nabla_{T_2} f(\mathbf{x}).$$

Therefore, we can write the sum of the attributions as follows:

$$\sum_{T_1 \sim \mathcal{U}(\mathcal{P}(\mathcal{I}_1))} |\nabla_{T_1} f(\mathbf{x})| = \sum_{T_2 \sim \mathcal{U}(\mathcal{P}(\mathcal{I}_2))} |\nabla_{T_2} f(\mathbf{x})|.$$

Since the attribution method (e.g., Integrated Directional Gradients) involves integrating the gradients over a path from the baseline $\mathbf{x}'$ to $\mathbf{x}$, we can express the attribution for each interaction set as:

$$\sum_{T_1 \sim \mathcal{U}(\mathcal{P}(\mathcal{I}_1))} \int_{\alpha=0}^{1} |\nabla_{T_1} f(\mathbf{x}(\alpha)| = \sum_{T_2 \sim \mathcal{U}(\mathcal{P}(\mathcal{I}_2))} \int_{\alpha=0}^{1} |\nabla_{T_2} f(\mathbf{x}(\alpha))|$$

Therefore,

$$a(\mathcal{I}_1) = a(\mathcal{I}_2)$$

□

F. Smoothing RELU

Instead of using the SoftPlus $s(z)$ like previous works [21], we use the following approximation of the ReLU $h(z)$ from Zhang *et al.* [52] as it provided less noisy explanations. Fig. 5 shows the plots for gradient and hessian for the functions with $\tau = 0.001$.

$$h(z) = \begin{cases} \left(z + \sqrt{z^2 + \tau}\right)' = 1 + \frac{z}{\sqrt{z^2+\tau}} & (z < 0) \\ \left(\sqrt{z^2 + \tau}\right)' = \frac{z}{\sqrt{z^2+\tau}} & (z \geq 0) \end{cases} \tag{8}$$

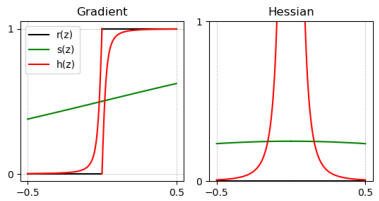


Figure 5. Gradient and Hessian comparison between the ReLU activation function $r(z)$, SoftPlus activation function $s(z)$, and Zhang *et al.* [52] $h(z)$

G. Higher-order feature interaction

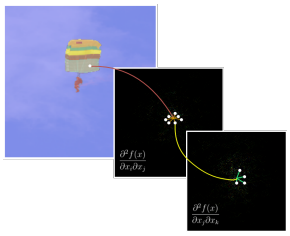


Figure 6. Diagram showing H-Sets algorithm. First, the image is segmented using SAM. Then, the highest attributed feature by Integrated Gradients in a mask is used as a starting feature to kick-start the algorithm. The Hessian matrix is then taken using this feature and the highest attributed features from the Hessian are appended to the feature interaction set. If more features can be added to the set, then the Hessian matrix of the highest attributed features from the previous Hessian is taken and the algorithm repeats recursively.

As discussed in Section 3.1.1, we propose to fix the starting feature x_i as the highest attributed feature by Integrated Gradients [44] in segmentation masks produced by Segment Anything Model (SAM) [24]. The overview of the H-Sets algorithm is shown in Figure 6.

H. Baselines

We evaluate our method against non-interaction-based attribution methods, Integrated Gradients [44], and

interaction-based attribution methods: CAFO, CASO [38], Archipelago [48] and MoXi [43].

H.1. Integrated Gradient

Integrated Gradients (IG) assigns importance scores to each input feature by integrating the model's gradients along a straight-line path from a baseline input $\mathbf{x}'$ (e.g., a black image or zero vector) to the actual input $\mathbf{x}$. This approach ensures that the attributions satisfy desirable properties such as sensitivity and implementation invariance.

Formally, given a model $f : \mathbb{R}^d \to \mathbb{R}$ and an input $\mathbf{x} \in \mathbb{R}^d$, the Integrated Gradient of the i-th input feature is defined as:

$$\text{IG}_i(\mathbf{x}) = (x_i - x_i') \int_{\alpha=0}^{1} \frac{\partial f(\mathbf{x}' + \alpha(\mathbf{x} - \mathbf{x}'))}{\partial x_i} \, d\alpha$$

where $\mathbf{x}'$ is the baseline input, and $\alpha \in [0, 1]$ is the interpolation parameter. In practice, the integral is approximated using a Riemann sum over m steps:

$$\text{IG}_i(\mathbf{x}) \approx (x_i - x_i') \cdot \frac{1}{m} \sum_{k=1}^{m} \frac{\partial f\left(\mathbf{x}' + \frac{k}{m}(\mathbf{x} - \mathbf{x}')\right)}{\partial x_i}$$

Integrated Gradients (IG) is one of the most widely used feature attribution methods. We use the implementation of IG available in the Captum library [25].

H.2. Context-Aware First-Order and Second-Order Attributions

To provide structured and faithful feature attributions, the Context-Aware First-Order (CAFO) and Context-Aware Second-Order (CASO) methods formulate feature importance as an optimization problem that explicitly incorporates group-feature perturbations under sparsity and smoothness constraints [39]. Given a trained model f_{θ^*}, an input-label pair (x, y), and the loss function $\ell(f_{\theta^*}(x), y)$, both CAFO and CASO aim to find a perturbation $\Delta = \tilde{x} - x$ that maximizes the change in loss while penalizing large and non-sparse perturbations. This formulation captures the idea that important feature subsets are those whose perturbation causes significant changes in model behavior.

The **CAFO** method linearizes the loss function around the input and solves the following optimization:

$$\tilde{I}^{\text{CAFO}}_{\lambda_1,\lambda_2}(x, y) = \max_{\Delta} \nabla_x \ell(f_{\theta^*}(x), y)^\top \Delta - \lambda_1 \|\Delta\|_1 - \lambda_2 \|\Delta\|_2^2 \tag{9}$$

where λ_1 and λ_2 are regularization parameters controlling the sparsity and magnitude of the perturbation.

To better approximate the true model behavior, the **CASO** method extends CAFO by using a second-order Taylor expansion of the loss function:

$$\tilde{I}^{\text{CASO}}_{\lambda_1,\lambda_2}(x, y) = \max_{\Delta} \nabla_x \ell(f_{\theta^*}(x), y)^\top \Delta + \frac{1}{2} \Delta^\top H_x \Delta - \lambda_1 \|\Delta\|_1 - \lambda_2 \|\Delta\|_2^2 \tag{10}$$

where H_x is the Hessian of the loss with respect to the input features. The inclusion of the curvature term $\Delta^\top H_x \Delta$ enables CASO to capture model curvature.

We use the author's official implementation available in Github [37].

H.3. Archipelago

Archipelago is a model-agnostic framework for detecting and attributing feature interactions [48]. It consists of two main components: ArchAttribute, an interaction-aware attribution method, and ArchDetect, a scalable interaction detection algorithm. Together, they generate axiomatic explanations by isolating groups of features—referred to as "feature islands"—whose combined contributions drive model predictions.

Given a black-box model f, an input instance $x^\star$, and a baseline x', ArchAttribute assigns an attribution score $\phi(I)$ to each feature set $I \subseteq [d]$, defined as:

$$\phi(I) = f(x^\star_I + x'_{\setminus I}) - f(x') \tag{11}$$

This formulation captures the isolated contribution of the feature group I by embedding it in a neutral baseline context x'. ArchAttribute is designed to be additive and satisfies several generalized axioms. To efficiently identify relevant interaction sets $\mathcal{S} = \{I_1, I_2, \dots\}$, Archipelago also introduces ArchDetect. Please see the paper [48] for more details.

We use the official implementation of Archipelago provided by the authors on GitHub [47]. For a fair comparison, we fix the number of feature interaction sets to 5 in Archipelago. We use the same number of interaction sets in our method.

H.4. MoXi

MoXI (Model eXplanation by Interactions) is a recent method that efficiently identifies and attributes groups of pixels using Shapley values and their interactions [43]. The framework provides two complementary approaches: *pixel insertion*, which measures the confidence gain when a pixel is revealed, and *pixel deletion*, which measures the confidence drop when a pixel is masked. In this paper, we employ the pixel insertion variant.

Because game-theoretic quantities such as Shapley values and interactions are computationally expensive, MoXI introduces a super-pixel representation and a greedy selection strategy. Let k denote the number of pixels to be added

to the interaction set $\mathcal{I}$, N the total number of pixels, and f the model. According to the pixel insertion algorithm, to add a pixel b_k to $\mathcal{I}$, the method selects

$$b_k \leftarrow \arg\max_{b \in N \setminus \mathcal{I}} f(\mathcal{I} \cup \{b\}),$$

repeating this process k times to construct the interaction set.

We use the original implementation of MoXI provided in their official GitHub repository [42].

I. Evaluation Metrics

Saliency maps only provide a qualitative evaluation of an explanation method. However, such evaluation does not provide an accurate comparison because of associated human bias and lack of "ground-truth" data. In this work, we focus on two important evaluation metrics: sparsity and faithfulness.

I.1. Sparsity

Sparsity evaluates the comprehensibility of explanations. We evaluate the sparsity of the attribution vector $\phi(\mathbf{x})$ by calculating its Gini index, as implemented by Chalasani *et al*. [6]. For an attribution vector $\phi(\mathbf{x}) \in \mathbb{R}^d$, we first sort the absolute values in non-decreasing order and then compute the Gini index using Eqn. 12.

$$G(\phi(\mathbf{x})) = 1 - 2\sum_{k=1}^{d} \frac{\phi(\mathbf{x})_{(k)}}{||\phi(\mathbf{x})||_1} \frac{d-k+0.5}{d} \tag{12}$$

This formula calculates a weighted sum of fractions, where each fraction represents the contribution of the k-th largest element to the overall sparsity. Larger elements receive higher weights, emphasizing their impact on sparsity. The Gini index ranges from $[0, 1]$: a value of 1 indicates perfect sparsity, where only one element in $\phi(\mathbf{x})$ is greater than zero, while a value of 0 indicates no sparsity, meaning all vector elements are equal to some positive value.

I.2. Faithfulness

Faithfulness measures whether the explanation truly reflects the underlying model behavior. We use the ROAD evaluation by Rong *et al*. [30] for faithfulness. ROAD evaluates model accuracy on a test set during an iterative process where the top k most important pixels are removed at each step. Pixel removal is performed using a noisy linear imputation technique to minimize the creation of out-of-distribution samples.

For our experiments, we use the MoRF (Most Relevant First) strategy from the ROAD implementation available in Quantus [17]. However, both the MoRF and LeRF (Least Significant Removal First) strategy provides the same results as discussed in the original paper.

In MoRF, given a model F and an input, an attribution method assigns importance values to each feature. The features are then ranked in descending order of importance. At each step, the top $k = 5$ most important features are removed, and model accuracy is assessed. A sharper accuracy drop indicates a more effective explanation.

We opted for ROAD over Insertion/Deletion [28] and ROAR [18] because Insertion/Deletion introduces artifacts, leading to distribution shifts in perturbed inputs, and ROAR requires costly model retraining.

$\text{ROAD}_{\text{AOPC}}$ score. In addition, we quantify the ROAD plot using area over the perturbation curve, computed as $\text{ROAD}_{\text{AOPC}} = \frac{1}{L+1}\sum_{k=1}^{L}\left\langle f(x^{(0)}) - f(x^{(k)})\right\rangle$ where, L represents the number of feature removal steps, and $f(x)$ is the classifier's output for the originally predicted class given the input x. The term $x^{(0)}$ corresponds to the unperturbed input image, while $x^{(k)}$ represents the image after k perturbation steps. Higher $\text{ROAD}_{\text{AOPC}}$ score represents a more faithful method.

J. Additional experiments

J.1. Non-interaction baselines

In addition to the baseline methods discussed in Sec. H, we evaluate H-Sets against DeepLift [32] and LRP [1].

DeepLIFT [32] is explains a model's prediction by decomposing the "difference-from-reference" of the output into contributions from each input feature. It operates by comparing the activations of the actual input x against a user-defined "reference" input x_0 (e.g., a neutral background or blurred image). By defining the difference $\Delta t = t - t_0$ for a target neuron t, the method assigns contribution scores $C_{\Delta x_i \Delta t}$ that satisfy the summation-to-delta property: $\sum C_{\Delta x_i \Delta t} = \Delta t$.

Layer-wise Relevance Propagation (LRP) [1] redistributes the prediction score (relevance) from the output layer back to the input pixels, following the fundamental conservation principle: the total relevance R must be preserved across each layer of the network ($\sum R_i = \sum R_j$). In this framework, each neuron distributes its relevance to its predecessors based on their relative contribution to its activation, using specific decomposition rules to handle different weight distributions.

Tab. 11 and Tab. 12 demonstrate the sparsity and faithfulness scores. Compared with the main results in Tabs. 1 and 2, these additional non-interaction baselines further reinforce the same overall pattern. In terms of sparsity, DeepLift and LRP are occasionally competitive with first-order baselines. However, this sparsity does not translate into consistently strong faithfulness. Across most archi-

Table 11. Sparsity comparison (Gini index; higher is better) of LRP and DeepLift. Values are mean ± std for 5 runs.

Dataset	Model	LRP	DeepLift
ImageNet	VGG	0.681±0.068	0.721±0.073
	ResNet	0.965±0.022	0.601±0.054
	DenseNet	0.627±0.078	0.611±0.077
	MobileNet	0.584±0.057	0.567±0.053
CUB	VGG	0.752±0.135	0.703±0.082
	ResNet	0.983±0.022	0.586±0.061
	DenseNet	0.690±0.170	0.556±0.061
	MobileNet	0.600±0.059	0.563±0.062

Table 12. Faithfulness comparison ($\text{ROAD}_{\text{AOPC}}$; higher is better) of LRP and DeepLift. Values are mean ± std for 5 runs.

Dataset	Model	LRP	DeepLift
ImageNet	VGG	0.357±0.023	0.431±0.073
	ResNet	0.231±0.042	0.224±0.051
	DenseNet	0.265±0.043	0.425±0.027
	MobileNet	0.271±0.034	0.348±0.056
CUB	VGG	0.565±0.073	0.566±0.085
	ResNet	0.375±0.021	0.437±0.054
	DenseNet	0.385±0.035	0.584±0.076
	MobileNet	0.529±0.094	0.448±0.073

tectures, their $\text{ROAD}_{\text{AOPC}}$ scores remain below those of **H-Sets**. In contrast, **H-Sets** maintains a stronger balance between sparsity and faithfulness across datasets and architectures.

J.2. Additional faithfulness metric

In addition to ROAD [30], we evaluate the faithfulness using faithfulness correlation [2]. It operates by iteratively selecting a random subset of features ($|S|$), replacing them with baseline values, and then calculating the Pearson's correlation coefficient between the average attribution of those features and the resulting change in the model's predicted logits. By averaging these correlations over multiple runs and test samples, the metric produces a correlation score, where higher values indicate a more "faithful" explanation.

Tab. 13 demonstrates that H-Sets consistently outperforms all other methods across datasets and networks on this metric.

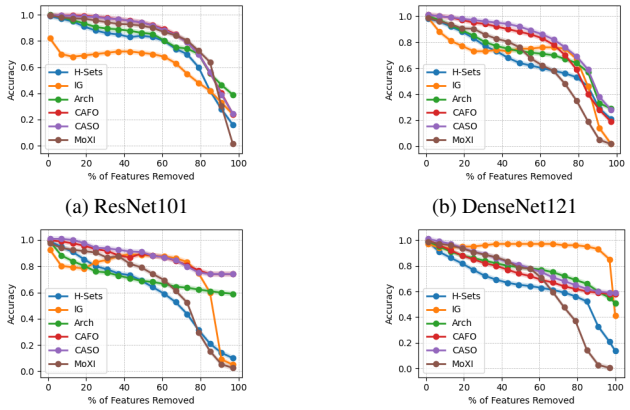
K. ROAD plots

K.1. ImageNet

Figure 7 shows the ROAD curves for the ImageNet dataset. These plots illustrate how model accuracy degrades as top-attributed features are progressively removed, providing a direct evaluation of attribution faithfulness.

Table 13. Faithfulness Correlation (higher is better) of our method (H-Sets) versus Integrated Gradients (IG) [44], context-aware first-order (CAFO), and second-order explanations (CASO) [38], Archipelago (Arch) [48], and MOXI [43]. Values are mean ± std for 5 runs.

		Explanation Methods					
Dataset	**Model**	IG	Arch	CAFO	CASO	MoXI	**H-Sets**
ImageNet	VGG	-0.017±0.106	0.009±0.108	-0.007±0.102	-0.001±0.101	0.007±0.119	**0.009±0.122**
	ResNet	-0.016±0.099	-0.044±0.381	0.004±0.109	0.004±0.118	-0.024±0.349	**0.251±0.101**
	DenseNet	-0.005±0.127	**0.059±0.147**	-0.037±0.121	-0.032±0.119	0.012±0.126	0.034±0.135
	MobileNet	-0.015±0.117	0.020±0.138	-0.001±0.105	0.000±0.104	0.011±0.129	**0.021±0.113**
CUB	VGG	-0.015±0.119	0.033±0.317	0.009±0.355	0.008±0.111	-0.009±0.136	**0.124±0.157**
	ResNet	-0.008±0.118	0.122±0.319	0.016±0.103	0.003±0.111	0.037±0.341	**0.123±0.426**
	DenseNet	-0.008±0.114	0.111±0.353	**0.144±0.340**	-0.021±0.108	0.025±0.153	0.029±0.126
	MobileNet	-0.004±0.119	0.035±0.303	0.092±0.302	-0.001±0.104	0.013±0.110	**0.128±0.119**



(a) ResNet101 (b) DenseNet121

(c) VGG16 (d) MobileNetV3

Figure 7. ROAD plots: ImageNet

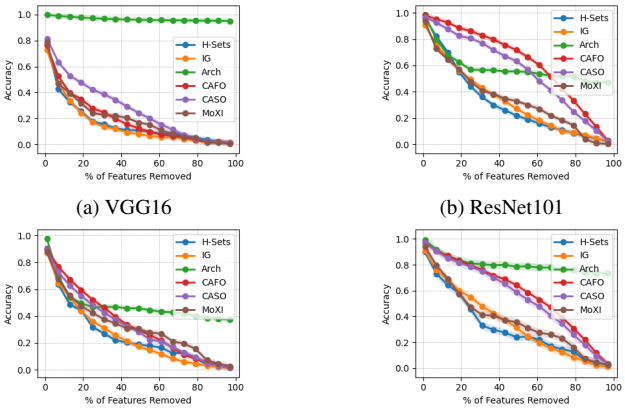
K.2. CUB

Figure 8 presents the **ROAD** curves for the CUB dataset [49], showing how model accuracy degrades as increasingly important features are progressively removed. These plots visually corroborate the $\text{ROAD}_{\text{AOPC}}$ scores reported in Table 2, where H-Sets achieves the highest scores across all models, reflecting the area over these perturbation curves.

L. Ablation: Hyperparameter (CUB)

We conduct an ablation study on the CUB dataset[49] using the ResNet101 [16] model to evaluate the robustness of H-Sets to two key hyperparameters: the number of features in the interaction set ν and the interaction threshold μ that determines the minimum pairwise Hessian strength.

Number of features. Table 14 (top) shows that using very few features (e.g., 250) yields highly sparse maps (high sparsity score), but these can underrepresent the model's behavior, especially on large images like CUB, where ob-



(a) VGG16 (b) ResNet101

(c) DenseNet121 (d) MobileNetV3

Figure 8. Faithfulness evaluation using ROAD plots for CUB across different models.

Table 14. Ablation of H-Sets hyperparameters on CUB dataset using MobileNetV3. **Top**: varying the number of features included in explanations ν. **Bottom**: varying the Hessian interaction threshold μ.

# Features (ν)	**Sparsity**	**ROAD$_{\text{AOPC}}$**
250	0.990	0.643
1000	0.961	0.629
2000	0.928	0.647
3000	0.898	0.611
5000	0.845	0.598
Threshold (μ)	**Sparsity**	**ROAD$_{\text{AOPC}}$**
0.1	0.903	0.614
0.2	0.903	0.596
0.3	0.901	0.609
0.4	0.901	0.609
0.5	0.898	0.610
0.6	0.903	0.599
0.7	0.900	0.615
0.8	0.971	0.607

jects typically span large spatial regions. As we increase the number of features, sparsity decreases steadily, which is expected as more interacting features are included. Despite this, the measure of faithfulness with ROAD$_{\text{AOPC}}$ score remains consistently high across the range, suggesting that our method reliably identifies relevant interactions. However, increasing the number of features raises computational cost, as our method performs Hessian-based interaction detection and set-level directional attributions. We find that selecting 2000–3000 features provides an effective trade-off: the explanations are still sparse and interpretable, ROAD scores remain high, and the computation remains tractable (see Figure 9 for sample explanations).

Interaction threshold. Table 14 (bottom) shows the effect of varying the interaction threshold μ, with the number of features fixed at 3000. We observe that both sparsity and ROAD$_{\text{AOPC}}$ scores remain stable across a wide range of μ values, indicating that H-Sets is robust to this hyperparameter. This robustness indicates that H-Sets reliably detects semantically meaningful interactions without being overly sensitive to the exact strength of second-order gradients. From a deployment perspective, this robustness simplifies hyperparameter tuning, making H-Sets practical to apply across datasets and architectures with minimal adjustment.

Figure 9. Saliency maps with different values of ν on CUB.