Making Sense of Dependence: Efficient Black-box Explanations Using Dependence Measure

Paul Novello [1 2] **Thomas Fel** [2 3] **David Vigouroux** [1 2]

[1] IRT Saint Exupery, France, [2] Artificial and Natural Intelligence Toulouse Institute, Université de Toulouse, France [3] Carney Institute for Brain Science, Brown University, USA
paul.novello@irt-saintexupery.com

Abstract

This paper presents a new efficient black-box attribution method based on Hilbert-Schmidt Independence Criterion (HSIC), a dependence measure based on Reproducing Kernel Hilbert Spaces (RKHS). HSIC measures the dependence between regions of an input image and the output of a model based on kernel embeddings of distributions. It thus provides explanations enriched by RKHS representation capabilities. HSIC can be estimated very efficiently, significantly reducing the computational cost compared to other black-box attribution methods. Our experiments show that HSIC is up to 8 times faster than the previous best black-box attribution methods while being as faithful. Indeed, we improve or match the state-of-the-art of both black-box and white-box attribution methods for several fidelity metrics on Imagenet with various recent model architectures. Importantly, we show that these advances can be transposed to efficiently and faithfully explain object detection models such as YOLOv4. Finally, we extend the traditional attribution methods by proposing a new kernel enabling an ANOVA-like orthogonal decomposition of importance scores based on HSIC, allowing us to evaluate not only the importance of each image patch but also the importance of their pairwise interactions. Our implementation is available at https://github.com/paulnovello/HSIC-Attribution-Method.

1 Introduction

Artificial Intelligence has established itself as the reference technique for tackling many real-world automation tasks. Consequently, the diversity of its applications is growing and reaching fields where its outputs can contribute to critical decision-making. In such cases, it is essential to be able to provide explanations for each link of the decision chain, including AI algorithms. Over the past decade, many techniques have emerged to explain the predictions of these algorithms [48, 40, 45, 17, 58, 36, 35, 30, 29, 42], marking the birth of a new field called Explainable Artificial Intelligence (XAI). The tools developed in this research field, mostly designed to explain neural networks, have already proven helpful. For instance, it has been used in model debugging, identification of new development strategies for practitioners, and failure understanding.

Initial approaches are based on analyzing the internal state of neural networks during inference, often relying on input gradients or activation values of hidden layers [48, 45, 17, 28]. However, the gradient only reflects the model's operation in an infinitesimal neighborhood around an input and can therefore be misleading [20]. Furthermore, their applicability is limited to the case where the final user has access to the implementation of the model. Therefore, such methods cannot be applied in the most common use cases, e.g. when models are made available by third parties through

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

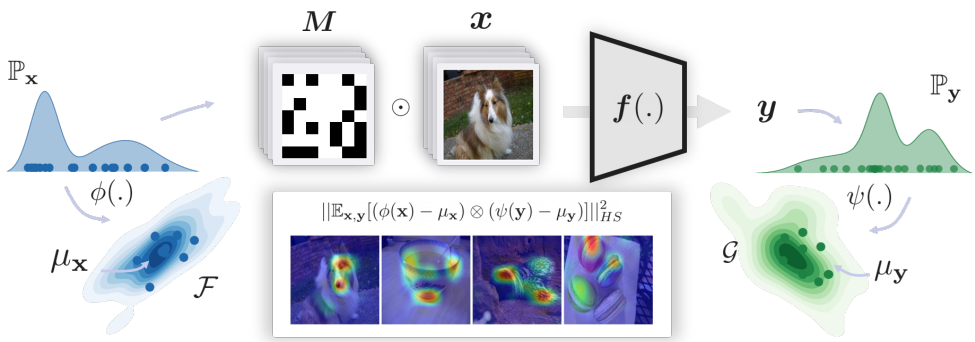

Figure 1: **HSIC Explainability method**. We sample random binary masks $\mathbf{M}$ that we use to perturb the input image $\boldsymbol{X}$. We obtain a perturbed output and measure the dependence between the distribution of each patch $\mathbb{P}_M$ of the binary mask and that of the output $\mathbb{P}_{\mathbf{y}}$. We use a dependence measure, Hilbert Schmidt Independence Criterion (HSIC), based on the kernel embedding of this distribution in a Reproducing Kernel Hilbert Space (RKHS). Each patch is then assigned the value of this measure: the more independent a patch is from $\mathbf{y}$, the less important it is to explain it.

API calls or specialized hardware. In order to address this issue, some black-box approaches have been recently proposed, relying on the perturbation of the input and the observation of its effect on the output [61, 40, 36, 13]. One challenge of such perturbation methods is assessing this effect together with taking into account complex interactions inherent to deep neural networks. To account for these characteristics, black-box methods resort to complex Monte Carlo methods that require a high number of model forward passes, which can be expensive for recent neural networks that are growing larger.

In [13, 32], the authors propose methods to reduce the required number of forward passes, but the obtained performance improvements still do not make them close to white-box methods. In this work, similarly to [13], we cast perturbation studies as Global Sensitivity Analysis (GSA) [38]. However, we rely on a whole different approach based on dependence measure rather than analysis of variance. We measure the dependence between patch-wise perturbations of an input image and the model's output by comparing the distribution of perturbed inputs and outputs embedded in a Reproducing Kernel Hilbert Space (RKHS). More specifically, we use Hilbert-Schmidt Independence Criterion (HSIC), a dependence measure based on the Hilbert-Schmidt norm of the empirical cross-covariance operator evaluated between the represented distribution. HSIC leverages the rich theory of RKHS, thereby capturing more diverse information than variance-based indices such as Sobol. In addition, it can be estimated more efficiently, even bridging the performance gap between black-box and white-box methods.

Our contributions are as follows: **(1)** we introduce a new efficient black-box attribution method relying on HSIC; **(2)** we derive a new kernel that confers an Analysis of Variance (ANOVA)-like orthogonal decomposition property, allowing us to go beyond usual attribution methods and evaluate interactions between patches of the image; **(3)** we conduct experiments to assess the fidelity of our method on ImageNet and show that it improves or matches the state-of-the art for different metrics while bridging the computational gap between black-box and white-box attribution methods; **(4)** we demonstrate its versatility and its potential by successfully applying it to a less common test case: explanations of object detection; and a new test case: evaluation of pairwise interactions between patches of the input image.

2 Related work

Our work builds on prior efforts aiming to develop attribution methods in order to explain the prediction of a deep neural network by pointing to input variables that support the prediction – typically pixels or groups of pixels, i.e. patches in the image – which lead to importance maps.

Attribution methods for white-box models A large number of attribution methods have been developed relying on the gradient of the decision studied. The first method was introduced in [4]

and improved in [48, 61, 56] and consists of explaining the decisions of a convolution model by back-propagating the gradient from the output to the input, indicating which pixels affect the decision score the most. However, this family of methods is limited because they focus on the influence of individual pixels in an infinitesimal neighborhood around the input image in the image space. For instance, it has been shown that gradients often vanish when the prediction score to be explained is near the maximum value [58]. Integrated Gradient [58] and SmoothGrad [50] partially address this issue by accumulating gradients. Another family of attribution methods relies on the neural network's activations. Popular examples include CAM [62], which computes an attribution score based on a weighted sum of feature channel activities – right before the classification layer. GradCAM [45] extends CAM via the use of gradients, reweighting each feature channel to take into account their importance for the predicted class. Nevertheless, the choice of the layer has a huge impact on the quality of the explanation. In comparison, our proposed approach is model-agnostic and hence does not require access to internal computations.

Attribution methods for black-box models In this paper, we extend the problem by restricting it to a black-box model: the analytical form and potential internal states of the model are unknown. Several methods compute influence scores for each individual pixel or group of pixels.

The first method, Occlusion [61], masks individual image regions – one at a time – with an occluding mask set to a baseline value and assigns the corresponding prediction scores to all pixels within the occluded region. Then the explanation is given by these prediction scores and can be easily interpreted. However, occlusion fails to account for the joint (higher-order) interactions between multiple image regions. For instance, occluding two image regions – one at a time – may only decrease the model's prediction minimally (say a single eye or mouth component on a face), while occluding these two regions together may yield a substantial change in the model's prediction if these two regions interact non-linearly as is expected for a deep neural network. Our work, together with related methods such as LIME [40], RISE [36] and more recently Sobol [13] addresses this problem by randomly perturbating the input image in multiple regions at a time.

Surprisingly, RISE [36] and Sobol Attribution [13] have recently shown that black-box attribution methods can rival and even surpass the white-box methods commonly used without recourse to internal states. However, despite the efforts in [13] to limit their computational overhead, black-box methods remain far from white-box methods in terms of execution time. In this work, we show that it is possible to match or even surpass the performances of current black-box methods while reaching computation times lower than some white-box methods by using dependence measure-based Global Sensitivity Analysis (GSA).

Global sensitivity analysis using dependence measures Our attribution method builds on the GSA framework. The approach was introduced in the 70s [9] and was popularized with variance-based sensitivity analysis and Sobol indices [51]. It consists of evaluating the sensitivity of a model's output of interest to some input design variables. GSA is currently used in many fields, especially for the study of physical phenomena [27, 38]. Recently, dependence measure-based sensitivity analysis was introduced in [10] and was shown to circumvent some practical issues of variance-based sensitivity analysis. In particular, by relying on the representation capabilities of RKHS, the dependence measure that we use in this work, HSIC [22], captures more diverse information than traditional variance-based indices for far fewer model evaluations.

3 Explanations using Hilbert-Schmidt Independence Criterion

In this section, we describe sensitivity analysis-based attribution methods, define Hilbert-Schmidt Independence Criterion (HSIC) [22] and explain how we can use it and adapt it to design a new black-box attribution method whose efficiency competes with white-box methods. We also explain the theoretical advantages of HSIC that we can build on to go beyond traditional attribution methods.

3.1 Sensitivity analysis of perturbed black-box models

Let $\boldsymbol{f} : \mathcal{X} = \mathcal{X}_1, ..., \mathcal{X}_n \to \mathcal{Y}$ be the model under study, $x_i \in \mathcal{X}_i$ the input variables and $\boldsymbol{y} = \boldsymbol{f}(x_1, ..., x_n) \in \mathcal{Y}$ the output value of the model $\boldsymbol{f}$. GSA studies the sensitivity of $\boldsymbol{y}$ to each input x_i by considering them as iid (independent and identically distributed) random variables and assessing

the link between their distribution and that of the output after an initial input sampling. Given an input vector $\boldsymbol{X} = (x_1, ..., x_n)$, a prediction $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{X})$ can thus be explained using sensitivity analysis by applying random perturbations $\mathbf{x} = (\mathrm{x}_1, ..., \mathrm{x}_n), \mathrm{x}_i \sim \mathbb{P}_{\mathcal{X}_i}$ of the original $\boldsymbol{X}$ and analyzing the importance of each x_i for explaining the variations of $\boldsymbol{y}$ - which is considered a random variable, $\mathbf{y} \sim \mathbb{P}_{\mathbf{y}}$.

For image data, the inputs x_i are pixels. However, pixel perturbations would only emphasize low level explanations. To obtain high level and meaningful explanations, we rather consider a random perturbation mask $\boldsymbol{M} = (M_1, ..., M_d) \in [0, 1]^d$. We upsample this mask using a Nearest Neighbor interpolation method to obtain $u(\boldsymbol{M}) \in [0, 1]^n$, a patch-perturbated vector that we apply on the input image $\boldsymbol{X}$ using a mask operator $\boldsymbol{\tau} : \mathcal{X} \times [0, 1]^d \to \mathcal{X}$. More specifically, we use the inpainting operator defined by $\boldsymbol{\tau}(\boldsymbol{X}, \boldsymbol{M}) = \boldsymbol{X} \odot u(\boldsymbol{M}) + (1 - u(\boldsymbol{M}))\mu$, with $\odot$ the Hadamard product and μ a baseline value (here, μ is a black image with all pixels' value $= 0$ [40, 61]). Hence, the mask $\mathbf{M}$ aggregates the patch-wise random perturbations M_i that are sampled independently for each patch (M_i are iid). In practice, the perturbations contained in the mask are binary perturbations, to simulate whether the information contained in the patch is kept in the image or not.. We thereby assess the effect of each image patch, represented by M_i, on the output.

The perturbation methodology thus consists of $\mathbf{(1)}$ sampling p masks $\{\boldsymbol{M}^{(1)}, ..., \boldsymbol{M}^{(p)}\}$ from $\mathbf{M} \sim \mathbb{P}_{\mathbf{M}}$ (with $\mathbb{P}_{\mathbf{M}} = \mathbb{P}_{M_1} \times ... \times \mathbb{P}_{M_p}$), $\mathbf{(2)}$ applying them to the original input vector, leading to p perturbed input vectors (e.g., partially masked images) $\{\boldsymbol{\tau}(\boldsymbol{X}, \boldsymbol{M}^{(1)}), ..., \boldsymbol{\tau}(\boldsymbol{X}, \boldsymbol{M}^{(p)})\}$ $\mathbf{(3)}$ computing the predictions $\{\boldsymbol{y}^{(1)}, ..., \boldsymbol{y}^{(p)}\} = \{\boldsymbol{f}(\boldsymbol{\tau}(\boldsymbol{X}, \boldsymbol{M}^{(1)})), ..., \boldsymbol{f}(\boldsymbol{\tau}(\boldsymbol{X}, \boldsymbol{M}^{(p)}))\}$ and $\mathbf{(4)}$ statistically assessing the effect of each mask M_i on $\boldsymbol{y}$ by estimating a sensitivity measure between each $\mathbb{P}_{M_i}$ and $\mathbb{P}_{\mathbf{y}}$ from the previous sampling. In this paper, we consider that the more independent M_i is from $\mathbf{y}$, the less important the corresponding image patch is to explain it [10]. In the following, we describe HSIC, a dependence measure, and how to use it in practice.

3.2 Hilbert-Schmidt Independence Criterion

Let $\mathbf{x}$ and $\mathbf{y}$ be two random variables of probability distribution $\mathbb{P}_{\mathbf{x}}$ and $\mathbb{P}_{\mathbf{y}}$ defined on $\mathcal{X}$ and $\mathcal{Y}$. HSIC measures the dependence between $\mathbb{P}_{\mathbf{x}}$ and $\mathbb{P}_{\mathbf{y}}$ based on their embedding in Reproducing Kernel Hilbert Space (RKHS). Let $\varphi : \mathcal{X} \to \mathcal{F}$ and $\psi : \mathcal{Y} \to \mathcal{G}$ two continuous feature mapping between $\mathcal{X}$, $\mathcal{Y}$, and two RKHS $\mathcal{F}, \mathcal{G}$, such that the inner product between the feature embeddings of $x, x' \in \mathcal{X}$ in $\mathcal{F}$ is given by the kernel $k(x, x') = \langle \varphi(x), \varphi(x') \rangle$ (and $l(y, y') = \langle \psi(y), \psi(y') \rangle$ for $y, y' \in \mathcal{Y}$). The cross-covariance operator $C_{\mathbf{xy}} : \mathcal{G} \to \mathcal{F}$ between the random variables $\mathbf{x}$ and $\mathbf{y}$ is defined in [19] and can be written.

$$C_{\mathbf{xy}} = \mathbb{E}_{\mathbf{xy}}[(\varphi(x) - \mu_{\mathbf{x}}) \otimes (\psi(y) - \mu_{\mathbf{y}})],$$

where $\mu_{\mathbf{x}} = \mathbb{E}_{\mathbf{x}}[\varphi(x)]$ and $\mu_{\mathbf{y}} = \mathbb{E}_{\mathbf{y}}[\psi(y)]$ are the mean embedding of $\mathbf{x}$ and $\mathbf{y}$ in $\mathcal{F}$ and $\mathcal{G}$. When $\mathcal{F}$ and $\mathcal{G}$ are universal RKHS on the compact domains $\mathcal{X}$ and $\mathcal{Y}$, then $\|C_{\mathbf{xy}}\|_{HS} = 0$ if and only if $\mathbf{x}$ and $\mathbf{y}$ are independent, where $\| \cdot \|_{HS}$ denotes the Hilbert Schmidt norm (see [23]). In [22], the authors define HSIC as $\|C_{\mathbf{xy}}\|^2_{HS}$, which can be written:

$$\begin{aligned} HSIC(\mathbf{x}, \mathbf{y}) =& \mathbb{E}_{\mathbf{x}, \mathbf{x}', \mathbf{y}, \mathbf{y}'}[k(x, x')l(y, y')] + \mathbb{E}_{\mathbf{x}, \mathbf{x}'}[k(x, x')]\mathbb{E}_{\mathbf{y}, \mathbf{y}'}[l(y, y')] \\ & - 2\mathbb{E}_{\mathbf{x}, \mathbf{y}}[\mathbb{E}_{\mathbf{x}'}[k(x, x')]\mathbb{E}_{\mathbf{y}'}[l(y, y')]], \end{aligned} \tag{1}$$

where $\mathbf{x}, \mathbf{x}'$ and $\mathbf{y}, \mathbf{y}'$ are pairwise iid. HSIC can also be expressed in terms of Maximum Mean Discrepancy (MMD), which is a distance between mean embeddings defined in an RKHS [57]. More specifically, let the product RKHS $\mathcal{P} = \mathcal{F} \times \mathcal{G}$ with kernel $v((x, y), (x', y')) = k(x, x')l(y, y')$. Then, $HSIC(\mathbf{x}, \mathbf{y}) = \gamma_v^2(\mathbb{P}_{\mathbf{x}}\mathbb{P}_{\mathbf{y}}, \mathbb{P}_{\mathbf{x},\mathbf{y}})$, where γ_v is the MMD operator on $\mathcal{P}$. HSIC thus measures the distance between $\mathbb{P}_{\mathbf{xy}}$ and $\mathbb{P}_{\mathbf{y}}\mathbb{P}_{\mathbf{x}}$ embedded in $\mathcal{P}$ [10]. Since $\mathbf{x} \perp \mathbf{y} \Rightarrow \mathbb{P}_{\mathbf{xy}} = \mathbb{P}_{\mathbf{y}}\mathbb{P}_{\mathbf{x}}$, the closer these distributions are, in the sense of the MMD, the more independent they are.

Thus, given a set of inputs $\{\boldsymbol{x}_1, ..., \boldsymbol{x}_p\}$ and the associated outputs $\{\boldsymbol{y}_1, ..., \boldsymbol{y}_p\}$, [22] shows that HSIC can be estimated by

$$\mathcal{H}^p_{\mathbf{x},\mathbf{y}} = \frac{1}{(p-1)^2} \operatorname{tr}(KHLH), \tag{2}$$

where $H, L, K \in \mathbb{R}^{p \times p}$ and $K_{ij} = k(x_i, x_j), L_{i,j} = l(y_i, y_j)$ and $H_{ij} = \delta(i = j) - p^{-1}$. Using this formula, $\mathcal{H}^p_{\mathbf{x},\mathbf{y}}$ can be computed with a $\mathcal{O}(p^2)$ complexity. In this work, the input variables M_i are the patch perturbations. Therefore, we compute $\mathcal{H}^p_{M_i,\mathbf{y}}$ i.e. the estimation of the HSIC

between a patch M_i and the output $\mathbf{y}$, for each patch ($i \in \{1, ..., d\}$), see Algorithm 1. We denote $\mathcal{H}_i^p := \mathcal{H}_{M_i,\mathbf{y}}^p$ for clarity. In the next section, we discuss the kernels k and l and show that we can obtain a valuable ANOVA-like orthogonal decomposition property for HSIC-based indices, allowing to assess interactions between input variables.

3.3 Orthogonalisation of HSIC to enable evaluation of interactions

One question of interest in explainability is the measurement of the importance of a specific group of variables. Indeed, it is notorious that neural networks are highly non-linear, and as it has been demonstrated in several works [16, 13], the effects of the groups of variables are far from being additive. Concretely, some areas of the image may only be important in interaction with other areas, affecting the output only when both areas are perturbed at the same time - as we shall see in Section 4.4 (for instance, for mustaches of a puma).

Let $\{\mathrm{x}_1, ..., \mathrm{x}_n\} \in \mathcal{X}^n$ be a set of n univariate random variables. For any subset $A = \{l_1, ..., l_{|A|}\} \subseteq \{1, ..., n\}$, we denote $\mathbf{x}_A = (\mathrm{x}_{l_1}, ..., \mathrm{x}_{l_{|A|}})$ the vector of input variables with indices in A. When using Sobol indices based GSA, it is possible to measure the interactions between variables using ANOVA decomposition. For HSIC, the corresponding decomposition property (which is not ANOVA since HSIC does not measure the variance) can be stated as follows:

Property 1 (Orthogonal decomposition property)**.** *The orthogonal decomposition property is fulfilled if:*

$$HSIC(\mathbf{x}, \mathbf{y}) = \sum_{A \subseteq \{1,...,n\}} HSIC_A, \tag{3}$$

where each term $HSIC_A$ is given by

$$HSIC_A = \sum_{B \subseteq A} (-1)^{|A|-|B|} HSIC(\mathbf{x}_B, \mathbf{y}),$$

and $HSIC(\mathbf{x}_B, \mathbf{y})$ is defined as in equation (1) with kernels l and k_A.

In Appendix A, we introduce an example to illustrate why this property is necessary in order to properly assess the importance of interactions. This property was lacking for dependence measure-based sensitivity analysis until the work of [11], which shows that when using HSIC, a specific choice of kernel k can enable this decomposition. For any choice of l and characteristic univariate kernel k, it is possible to obtain the orthogonal decomposition property by defining the input kernel k_A such that

$$k_A(\boldsymbol{x}_A, \boldsymbol{x}'_A) = \prod_{i \in A} (1 + k_0(x_i, x'_i)), \;\; s.t. \;\; k_0(x, x') = k(x, x') - \frac{\int k(x,t)dP(t) \int k(x',t)dP(t)}{\int \int k(s,t)dP(s)dP(t)} \tag{4}$$

These conditions can be stringent, especially the right one, which implies to compute integrals (analytically or empirically). It can be non trivial for continuous input distributions p_x and classical kernel choices such as Radial Basis Function (RBF) of bandwidth σ, $k(x, x') = \exp(\|x - x'\|/2\sigma^2)$. This condition is alleviated when using discrete input variables of known densities, for which the integrals can be computed analytically. In particular, in this work, we rely on Proposition 1:

Proposition 1. *Let x_i be a Bernoulli variable of parameter $p = \frac{1}{2}$, and $\delta(x = x')$ the Dirac kernel. Then the following kernel k_A satisfies the decomposition property:*

$$k_A(\boldsymbol{x}_A, \boldsymbol{x}'_A) = \prod_{i \in A} (1 + k_0(x_i, x'_i)), \;\; s.t. \;\; k_0(x, x') = \delta(x = x') - \frac{1}{2}. \tag{5}$$

The proof is in Appendix A. As a practical consequence, if we sample binary masks from a Bernoulli variable of parameter $p = 1/2$, i.e. $M_i \sim B(p)$ for $i \in \{1, ..., d\}$, and use the kernel defined in equation (5), we can assess not only the importance of each patch in the image but also the importance of the interactions between patches. It allows to go beyond classical attribution methods and reveal areas of the image that are only important in interaction with other areas, i.e. that affect the output only when both areas are perturbated at the same time. Concretely, for two image patches indexed by i and j, the interaction HSIC, $\mathcal{H}_{i \times j}$, can be obtained with [11]:

$$\mathcal{H}_{i \times j}^p = \mathcal{H}_{(M_i, M_j),\mathbf{y}}^p - \mathcal{H}_{M_i,\mathbf{y}}^p - \mathcal{H}_{M_j,\mathbf{y}}^p. \tag{6}$$

We insist on the fact that if the decomposition property is not valid, subtracting $\mathcal{H}^p_{M_i,\mathbf{y}}$ and $\mathcal{H}^p_{M_j,\mathbf{y}}$ to $\mathcal{H}^p_{(M_i,M_j),\mathbf{y}}$ does not ensure that we assess the importance of the interactions only. Traces of the independent importance of M_i and M_j may remain in the obtained metric. Some qualitative benefits of such a property are illustrated in Section 4.4. This decomposition holds for any choice of kernel $l : \mathcal{Y} \to \mathcal{G}$. Therefore, in the following, we use the RBF kernel, with the common practice of choosing the bandwidth as the median of the output [10, 22, 54].

3.4 Sample efficiency of HSIC estimator

Several types of metrics are classically used for sensitivity analysis. The most famous one, Sobol indices [52] and its variants [18, 6] classically require p^2 model evaluations [25] to reach an estimation error of $\mathcal{O}(\frac{1}{\sqrt{p}})$. Recent design of experiments managed to reduce this requirement to $p \times (d+2)$ [43] (Theorem 1), but with the increase in complexity of state-of-the-art architectures and the high dimensionality of inputs (although mitigated by the use of d patches instead of all pixels), it can still be cumbersome. Despite this drawback, [13] uses Sobol indices to obtain explanations and still achieves execution time improvement compared to other state-of-the-art attribution methods, which are even less efficient in terms of samples requirements.

HSIC is much less expensive to estimate than Sobol indices: for a same estimation error of $\mathcal{O}(\frac{1}{\sqrt{p}})$, p forward passes are needed instead of $p \times (d+2)$ [22] (Theorem 1). This allows using far fewer samples to obtain relevant explanations, thereby dramatically increasing the efficiency of the method compared to previous black-box approaches. This huge advantage is empirically illustrated in Section 4.2, where we demonstrate that our method defines a new standard in terms of efficiency for black-box attribution methods. It even bridges the efficiency gap between black-box and white-box approaches.

3.5 Implementation of the method

A summary of the whole attribution method is provided in Algorithm 1. The computation of $\mathcal{H}^p_i$ is $O(p^2)$ but it is possible to vectorize it using any library optimized for tensor operations (e.g. tensorflow). As a result, the computation time of $\mathcal{H}^p_i$ is negligible compared to that of the p forward passes. Furthermore, we implemented a sampling based on Latin Hypercube Sampling [33], a Quasi-Monte Carlo (QMC) method designed to efficiently fill the input space in Monte Carlo integration. Once the grid of $\mathcal{H}^p_i$ is obtained, we use a bilinear upsampling to be able to apply it to the image.

Algorithm 1 Explanations using HSIC-based sensitivity analysis as attribution method

1: **Inputs:** d the dimension of the masks, p the number of forward pass, $\boldsymbol{X}$ an input image.
2: Sample p binary masks $\{\boldsymbol{M}^{(1)}, ..., \boldsymbol{M}^{(p)}\}$ using LHS.
3: Compute the perturbed inputs $\{\boldsymbol{\tau}(\boldsymbol{X}, \boldsymbol{M}^{(1)}), ..., \boldsymbol{\tau}(\boldsymbol{X}, \boldsymbol{M}^{(p)})\}$
4: Compute the predictions $\{\boldsymbol{y}^{(1)}, ..., \boldsymbol{y}^{(p)}\} = \{\boldsymbol{f}(\boldsymbol{\tau}(\boldsymbol{X}, \boldsymbol{M}^{(1)})), ..., \boldsymbol{f}(\boldsymbol{\tau}(\boldsymbol{X}, \boldsymbol{M}^{(p)}))\}$
5: **for** $i \in \{1, ..., d\}$ **do**
6: Compute $\mathcal{H}^p_i$ using equation (2) and assign this value to the i-th patch of the input image.

4 Experiments

This section showcases the benefits of our approach compared to other attribution methods. These benefits are threefold. First, the computational cost of HSIC attribution method is significantly lower than previous state-of-the-art methods, even bridging the performance gap between black-box and white-box methods. Second, our method improves state-of-the-art for several fidelity metrics, for black-box as well as white-box methods. Finally, the orthogonal decomposition property of HSIC allows to go beyond usual attribution methods and assess interactions between image patches.

In Section 4.1, we compute explanations of the predictions in the ILSVRC-2012 [12] classification task (ImageNet), for four common architectures, namely MobileNet [44], ResNet50 [24], EfficientNet [59] and VGG16 [49]. Then, we compare those explanations with these of other state-of-the-art black-box and white-box attribution methods in terms of fidelity and efficiency. In Section 4.2, we investigate the convergence of our method by measuring its correlation with a high sample estimator

	Method	*ResNet50*	*VGG16*	*EfficientNet*	*MobileNetV2*	Exec. time (s)
Del. (↓)						
White-box	Saliency [48]	0.158	0.120	0.091	0.113	0.360
	Grad.-Input [47]	0.153	0.116	0.084	0.110	0.023
	Integ.-Grad. [58]	0.138	**0.114**	**0.078**	0.096	**1.024**
	SmoothGrad [50]	0.127	0.128	0.094	**0.088**	0.063
	GradCAM++ [45]	**0.124**	0.125	0.112	0.106	0.127
	VarGrad [45]	0.134	0.229	0.224	0.097	0.097
Black-box	LIME [41]	0.186	0.258	0.186	0.148	6.480
	Kernel Shap [32]	0.185	0.165	0.164	0.149	4.097
	RISE [36]	0.114	0.106	0.113	0.115	8.427
	Sobol [13]	0.121	0.109	0.104	0.107	5.254
	$\mathcal{H}_i^p$ eff. (ours)	**0.106**	**0.100**	**0.095**	**0.094**	**0.956**
	$\mathcal{H}_i^p$ acc. (ours)	**0.105**	**0.099**	**0.094**	**0.093**	1.668
Ins. (↑)						
White-box	Saliency [48]	0.357	0.286	0.224	0.246	0.360
	Grad.-Input [47]	0.363	0.272	0.220	0.231	0.023
	Integ.-Grad. [58]	0.386	0.276	0.248	0.258	**1.024**
	SmoothGrad [50]	0.379	0.229	0.172	0.246	0.063
	GradCAM++ [45]	0.497	**0.413**	**0.316**	0.387	0.127
	VarGrad [45]	**0.527**	0.241	0.222	**0.399**	0.097
Black-box	LIME [41]	0.472	0.273	0.223	0.384	6.480
	Kernel Shap [32]	0.480	0.393	0.367	0.383	4.097
	RISE [36]	**0.554**	**0.485**	**0.439**	**0.443**	8.427
	Sobol [13]	0.370	0.313	0.309	0.331	5.254
	$\mathcal{H}_i^p$ eff. (ours)	0.470	0.387	0.357	0.381	**0.956**
	$\mathcal{H}_i^p$ acc. (ours)	0.481	0.395	0.366	0.392	1.668

Table 1: **Deletion** and **Insertion** scores obtained on 1,000 ImageNet validation set images (For Deletion, lower is better and for Insertion, higher is better). The execution times are averaged over 100 explanations of ResNet50 predictions with an RTX Quadro 8000 GPU. The first and second best results are **bolded** and underlined.

and comparing it with RISE [36] and Sobol [13]. In the remaining sections, we conduct additional experiments that show the versatility of our method. In Section 4.3, we evaluate HSIC attribution method to explain object detection on COCO dataset [31] with YOlOv4 [39]. We conclude the experiments with Section 4.4, where we showcase the use of the HSIC orthogonal decomposition property to assess interactions between image patches.

4.1 Fidelity of classification explanations

In this section, we evaluate the fidelity of the explanations with three fidelity metrics. The first, Deletion [36], assumes that the more faithful an explanation is, the quicker the prediction score should drop when pixels that are considered important are shut down. The second one, Insertion [36], instead adds pixels on a baseline image, starting with pixels that are associated with the highest importance scores of the explanation. Finally, μFidelity [5] creates random pixels subsets which are assigned a baseline value and measure the correlation between the drop in the score and the importance of the explanation. Those metrics are further described in Appendix F.

In Table 1, we report the results of several different attribution methods for explaining the classification of MobileNet [44], ResNet50 [24], EfficientNet [59] and VGG16 [49] on 1000 images sampled from the ImageNet validation dataset. The models used for the experiments have been accessed from tensorflow [1] with the keras API [8]. We introduce two variants of our method, $\mathcal{H}_i^p$ eff. and $\mathcal{H}_i^p$ acc. The words "eff" and "acc" stand for efficient and accurate because we use $p = 764$ and $p = 1536$ samples, respectively. We use our method with a grid size of 7×7 ($d = 49$). To evaluate the different methods, we use the Xplique [15], a library dedicated to explainability. For black-box and white-box methods, we **bold** the best result and underline the second. When the differences between some methods are not statistically significant, we highlight both. Note that for μFidelity, the estimation

variance is high (typically about 20%), so we only use the bold notation and leave the Table in Appendix B. The exact error bars are also left in the Appendix to make the presentation lighter.

For the Deletion metric, our method obtains the best results among the tested black-box methods for all the architectures in both its efficiency and accurate variants. Except for EfficientNet, we even beat all tested white-box methods. RISE is still the best of black-box methods for the Insertion metric, but the accurate variant is systematically second. Besides, our methods are among the best of both black-box and white-box methods.

While HSIC is systematically better in Deletion, we can note that RISE overshadows it in Insertion. This could be explained by how these metrics are constructed. Deletion and Insertion metrics consist in measuring Area Under the Curve (AUC) of scores that respectively decrease and increase when deleting and adding patches, starting from a baseline image (see Appendix F for a detailed definition). Since Deletion measures a drop in the score starting from the original image, the faster the score drops, the better the metric. Hence, Deletion will favor methods that sharply identify important regions. On the contrary, since Insertion starts from an arbitrary baseline image, if the explanation map is more spread out, more relevant secondary information will be added, so the score will be better. As we can see in the maps of Appendix C, RISE saliency maps are way more spread out than HSIC's, which are sharper. It may explain why RISE is better in the Insertion benchmark and why HSIC attribution method dominates the Deletion benchmark. We provide additional quantitative examples to illustrate this link in Appendix F. Note that even if RISE dominates Insertion, it is far behind in Deletion. This is not the case for HSIC, which is still competitive in Insertion while dominating Deletion.

These results, as such, are already satisfactory. But it goes even further: we obtained these results with far fewer forward passes than other state-of-the-art black-box attribution methods. With the efficient variants, competitive results are obtained more than 8 times faster than RISE, the current standard of black-box attribution methods. It improves on Sobol, a recent and promising attribution method that was already branded as more efficient, by a factor 5. The time improvement factors for the accurate variant of our method are still very appealing (5 and 3). We even beat the execution time of Integrated Gradients white-box method [58], a popular and successful white-box method. The efficiency of HSIC attribution method is investigated more thoroughly in the next section.

4.2 Estimator efficiency

The advantage of black-box attribution methods lies in providing explanations without access to the model's internal state or the gradients. However, this advantage comes at a cost since many forward passes are needed to obtain meaningful explanations. This cost is all the more constraining since recent architectures are increasingly heavy in terms of computational time.

Therefore, it is critical for such attribution methods to use as few forward passes as possible. Results reported in Section 4.1 attest that our approach based on $\mathcal{H}_i^p$ shines in that regard, and we refer to this section for more comments. It motivates us to study the efficiency of our method further. To that end, we compute an "asymptotical"[1] explanation with $13,000$ forward passes, for HSIC, Sobol and RISE attribution methods, on 100 images of ImageNet validation set.

We apply the three methods on EfficientNet with $d = 7 \times 7$ masks and image patches, like in [36] and [13]. We then compute explanations for an increasing number of forward passes and compare the obtained explanation with the baseline "asymptotical" explanation. We use Spearman rank correlation [55] like theoretically and empirically argued in [21, 2, 60, 14]. This experiment allows comparing the convergence speed of our method with RISE and Sobol. The curves are plotted in Figure 2

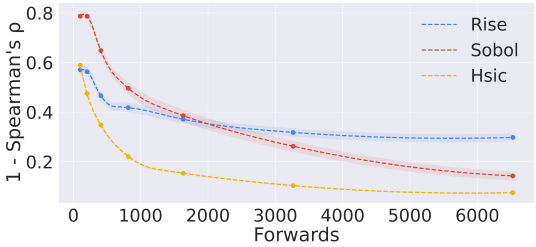

Figure 2: Spearman rank correlation of HSIC, Sobol and RISE attribution methods explanations on 100 ImageNet validation images with an "asymptotical" explanation based on $13,000$ samples.

[1]This explanation is not theoretically asymptotical (hence the quotation marks), but we use this designation because it is obtained with a very high number of forward passes

and show that our method, HSIC converges much faster than RISE and Sobol.

4.3 Explanation of object detection

Explaining model's predictions is more challenging for object detection than for image classification. Indeed, recent object detection models usually predict three pieces of information: localization (bounding box corners), objectness score (probability that a bounding box contains an object of any class), classification information (probability of each different possible class). Recently, it has been demonstrated that it was possible to use attribution methods to explain object detection by constructing a score aggregating for the previous information. In [37] the authors combine *intersection over union* for localization, *cosine similarity* for the class probability, and focus on high objectness areas. As a result, they can use RISE to explain the object detection using this score as the output of the model.

In this section, we test our method for explaining the object detections of YOLOv4 [39] on COCO dataset [31] compared to the approach presented in [37], D-RISE. We also compare the explanations with these of Kernel Shap [32], another black-box attribution method. The explanations for $1,000$ validation images are evaluated with the Deletion, Insertion, and μFidelity metrics. This experiment is time-consuming, so we use $\mathcal{H}_i^p$ eff. and 5000 samples for D-RISE and Kernel Shap.

Method	Deletion (↓)	Insertion (↑)	μFidelity (↑)	Exec. time (s)
D-RISE [37]	0.074	0.634	0.442	155
Kernel Shap. [32]	**0.070**	0.646	0.476	192
$\mathcal{H}_i^p$ (ours)	0.088	**0.658**	**0.568**	**34**

Table 2: Fidelity metrics obtained from explanations of YOLOv4 object detections on $1,000$ images of COCO validation data set. Execution times are averaged on the $1,000$ images on RTX 3080 GPUs.

Even if our method is not the best for Deletion, it is for Insertion and μFidelity, and more importantly, it is 5 and 6 times faster than D-RISE and Kernel Shap. Figure 3 displays visualizations of object detection explanations. While the first images show a standard detection explanation, the rightmost one is more interesting since it emphasizes an error of the object detector. The model identifies a zebra as a cat, and our method manages to explain this error by emphasizing the cat at the bottom right corner of the image. Note that we did not obtain such an explanation with D-RISE, even with a high sample number and different grid sizes - visualizations can be found in Appendix C).

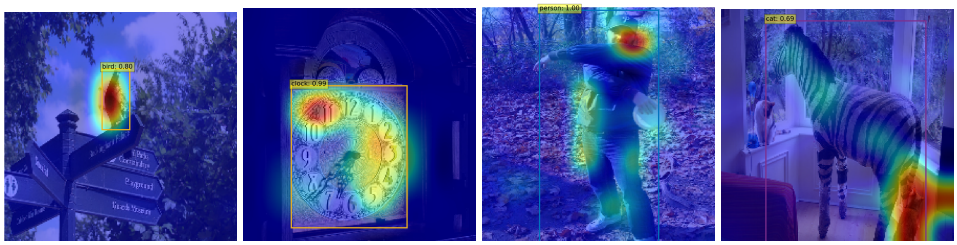

Figure 3: Visualisations of object detection explanations. The first three images show standard explanations, while the bottom right explains a misclassification. The zebra has been detected as a cat, and our method manages to explain why by emphasizing the cat at the bottom right corner.

4.4 Finding spacial interactions in the model

Usual attribution methods provide explanations in the form of heat maps that assign each pixel (or patch) an importance score. However, the scope of such explanations is limited since the reason for a prediction may not be explained only by the single importance of independent patches. In [13], the authors use Sobol total indices that account for the importance of a patch in interaction with all other patches, but they cannot localize the interactions. Thanks to its orthogonal decomposition property, our HSIC-based attribution method is able not only to assess the importance of each patch, but also the importance of interactions between specific patches by computing the HSIC of the joint patches

and subtracting the contribution of each patch taken independently[2]. It is then possible to identify regions of the image that affect the output only when both areas are perturbated jointly.

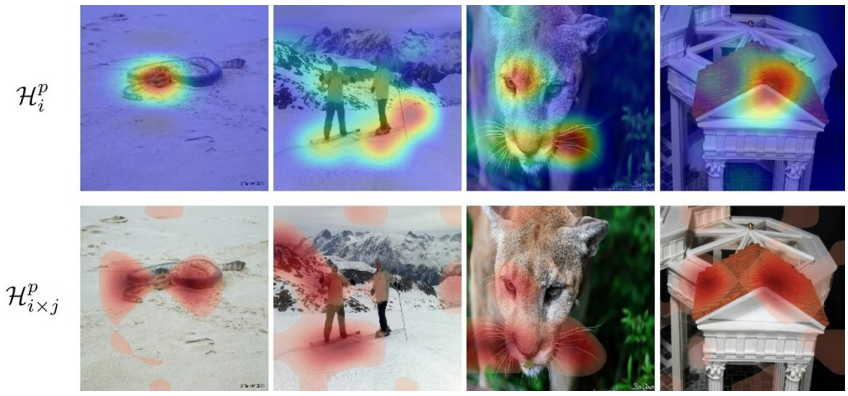

Figure 4: Upper row: $\mathcal{H}_i^p$, estimated HSIC for each image patch. Bottom row: highest $\mathcal{H}_{i\times j}^p$ between images patches. For the classification of the puma, the eye / forehead is the most important independent patch, but the right and left mustaches' interactions are even more important.

We illustrate this property in Figure 4. For each image, we computed all the possible interactions between patches and reported the most important ones. Note that not all images exhibit significant interactions, so we performed a qualitative selection for pedagogical purposes. On the upper row, we plot usual heatmaps that traduce the importance of each patch taken independently. On the bottom row, important interactions are represented in red. We can see that the middle part of the snake body is not the only important element in the picture, that one part of the mountain interacts with one of the skiers, so do the mustaches of the puma and two corners of the roof. Note that for each image, the maximum values of $\mathcal{H}_i^p$ are $40.6, 11.4, 8.2$ and 18.8 and the maximum values of $\mathcal{H}_{i\times j}^p$ are $19.6, 6.6, 9.2$ and 6.3 respectively. Thanks to the orthogonal decomposition property, these metrics can be compared and we can deduce that some interactions are as significant as some important independent patches. For the image with a puma, the interactions between the mustaches are even more important than the eye / forehead for identifying the animal ($\mathcal{H}_{i\times j}^p = 9.2$ when i and j are the right and left mustaches and $\mathcal{H}_{iff}^p = 8.2$ when i is the eye / forehead).

5 Conclusion

We have introduced a new attribution method based on a dependence measure, Hilbert-Schmidt Independence Criterion, which leverages representation capabilities of Reproducing Kernel Hilbert Spaces, thus being able to capture complex information. This attribution method is black-box, so it is applicable even when the implementation of the neural network to explain is not available. Nonetheless, it alleviates the computational burden of traditional black-box methods, improving on the state-of-the-art of both black-box and white-box attribution methods while being closer to the latter than the former in terms of efficiency. In addition, we showed how the rich framework of RKHS could be used to assess and localize interactions between pairs of patches of the input image that are relevant for explaining the output. We hope that the introduced framework will open up research avenues for attribution methods beyond traditional pixel-wise or patch-wise explanations.

Acknowledgements

This work has benefited from the AI Interdisciplinary Institute ANITI, which is funded by the French "Investing for the Future – PIA3" program under the Grant agreement ANR-19-P3IA-0004. The authors gratefully acknowledge the support of the DEEL project.[3]

[2]With the decomposition property, it is also possible to obtain HSIC "total" indices, like for Sobol, but it did not bring significant qualitative or quantitative advantages.

[3]`https://www.deel.ai/`

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] J. Adebayo, J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, and B. Kim. Sanity checks for saliency maps. *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[3] M. Ancona, E. Ceolini, C. Öztireli, and M. Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[4] D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K.-R. Müller. How to explain individual classification decisions. *The Journal of Machine Learning Research*, 11:1803–1831, 2010.

[5] U. Bhatt, A. Weller, and J. M. F. Moura. Evaluating and aggregating feature-based model explanations. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2020.

[6] E. Borgonovo. A new uncertainty importance measure. *Reliability Engineering & System Safety*, 92(6):771 – 784, 2007.

[7] A. Chattopadhay, A. Sarkar, P. Howlader, and V. N. Balasubramanian. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In *wacv*, 2018.

[8] F. Chollet et al. Keras. `https://keras.io`, 2015.

[9] R. Cukier, C. Fortuin, K. E. Shuler, A. Petschek, and J. Schaibly. Study of the sensitivity of coupled reaction systems to uncertainties in rate coefficients. i theory. *The Journal of chemical physics*, 59(8):3873–3878, 1973.

[10] S. Da Veiga. Global sensitivity analysis with dependence measures. *Journal of Statistical Computation and Simulation*, 85(7):1283–1305, May 2015.

[11] S. Da Veiga. Kernel-based anova decomposition and shapley effects–application to global sensitivity analysis. *arXiv preprint arXiv:2101.05487*, 2021.

[12] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[13] T. Fel, R. Cadene, M. Chalvidal, M. Cord, D. Vigouroux, and T. Serre. Look at the Variance! Efficient Black-box Explanations with Sobol-based Sensitivity Analysis. *arXiv:2111.04138 [cs]*, Nov. 2021.

[14] T. Fel and D. Vigouroux. Representativity and consistency measures for deep neural network explanations. *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2022.

[15] Fel, Thomas and Hervier, Lucas. Xplique: an neural networks explainability toolbox, 2021.

[16] G. Ferrettini, E. Escriva, J. Aligon, J.-B. Excoffier, and C. Soulé-Dupuy. Coalitional strategies for efficient individual prediction explanation. *Information Systems Frontiers*, pages 1–27, 2021.

[17] R. C. Fong and A. Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[18] J.-C. Fort, T. Klein, and N. Rachdi. New sensitivity analysis subordinated to a contrast. *Communications in Statistics - Theory and Methods*, 45(15):4349–4364, 2016.

[19] K. Fukumizu, F. R. Bach, and M. I. Jordan. Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. *J. Mach. Learn. Res.*, 5:73–99, dec 2004.

[20] S. Ghalebikesabi, L. Ter-Minassian, K. DiazOrdaz, and C. C. Holmes. On locality of local explanation models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[21] A. Ghorbani, A. Abid, and J. Zou. Interpretation of neural networks is fragile. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2017.

[22] A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. In *Proceedings of the 16th International Conference on Algorithmic Learning Theory*, ALT'05, page 63–77, Berlin, Heidelberg, 2005. Springer-Verlag.

[23] A. Gretton, R. Herbrich, A. Smola, O. Bousquet, and B. Scholkopf. Kernel methods for measuring independence. *Journal of Machine Learning Research*, 6(70):2075–2129, 2005.

[24] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *Proceedings of the IEEE European Conference on Computer Vision (ECCV)*, 2016.

[25] T. Homma and A. Saltelli. Importance measures in global sensitivity analysis of nonlinear models. *Reliability Engineering & System Safety*, 52(1):1–17, 1996.

[26] S. Hooker, D. Erhan, P.-J. Kindermans, and B. Kim. A benchmark for interpretability methods in deep neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2019.

[27] B. Iooss and P. Lemaître. A review on global sensitivity analysis methods. *Uncertainty management in Simulation-Optimization of Complex Systems: Algorithms and Applications*, 2015.

[28] A. Kapishnikov, T. Bolukbasi, F. Viégas, and M. Terry. Xrai: Better attributions through regions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[29] B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*. Proceedings of the International Conference on Machine Learning (ICML), 2018.

[30] P. W. Koh and P. Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR, 2017.

[31] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *Proceedings of the IEEE European Conference on Computer Vision (ECCV)*. Springer, 2014.

[32] S. M. Lundberg and S.-I. Lee. A unified approach to interpreting model predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 4768–4777, Red Hook, NY, USA, 2017. Curran Associates Inc.

[33] M. D. McKay. Latin hypercube sampling as a tool in uncertainty analysis of computer models. In *Proceedings of the 24th Conference on Winter Simulation*, WSC '92, page 557–564, New York, NY, USA, 1992. Association for Computing Machinery.

[34] P. Novello, G. Poëtte, D. Lugato, and P. Congedo. Goal-oriented sensitivity analysis of hyperparameters in deep learning. 2021.

[35] C. Olah, A. Mordvintsev, and L. Schubert. Feature visualization. *Distill*, 2017.

[36] V. Petsiuk, A. Das, and K. Saenko. Rise: Randomized input sampling for explanation of black-box models. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2018.

[37] V. Petsiuk, R. Jain, V. Manjunatha, V. I. Morariu, A. Mehra, V. Ordonez, and K. Saenko. Black-box explanation of object detectors via saliency maps. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11438–11447, 2021.

[38] S. Razavi, A. Jakeman, A. Saltelli, C. Prieur, B. Iooss, E. Borgonovo, E. Plischke, S. Lo Piano, T. Iwanaga, W. Becker, S. Tarantola, J. H. Guillaume, J. Jakeman, H. Gupta, N. Melillo, G. Rabitti, V. Chabridon, Q. Duan, X. Sun, S. Smith, R. Sheikholeslami, N. Hosseini, M. Asadzadeh, A. Puy, S. Kucherenko, and H. R. Maier. The future of sensitivity analysis: An essential discipline for systems modeling and policy support. *Environmental Modelling & Software*, 137:104954, 2021.

[39] J. Redmon and A. Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

[40] M. T. Ribeiro, S. Singh, and C. Guestrin. "why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016.

[41] M. T. Ribeiro, S. Singh, and C. Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 1135–1144, New York, NY, USA, 2016. Association for Computing Machinery.

[42] A. Ross, H. Lakkaraju, and O. Bastani. Learning models for actionable recourse. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[43] A. Saltelli. Making best use of model evaluations to compute sensitivity indices. *Computer Physics Communications*, 145(2):280–297, May 2002.

[44] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[45] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[46] A. Shrikumar, P. Greenside, and A. Kundaje. Learning important features through propagating activation differences. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2017.

[47] A. Shrikumar, P. Greenside, A. Shcherbina, and A. Kundaje. Not just a black box: Learning important features through propagating activation differences. *arXiv preprint arXiv:1605.01713*, 2016.

[48] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *Workshop Proceedings of the International Conference on Learning Representations (ICLR)*, 2014.

[49] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

[50] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg. Smoothgrad: removing noise by adding noise. In *Workshop on Visualization for Deep Learning, ICML*, 2017.

[51] I. M. Sobol. Sensitivity analysis for non-linear mathematical models. *Mathematical modelling and computational experiment*, 1:407–414, 1993.

[52] I. M. Sobol. Sensitivity estimates for nonlinear mathematical models. *MMCE*, 4(1):407–414, 1993.

[53] M. Sotoudeh and A. V. Thakur. Computing linear restrictions of neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2019.

[54] A. Spagnol, R. L. Riche, and S. Da Veiga. Global sensitivity analysis for optimization with variable selection. *SIAM/ASA J. Uncertain. Quantification*, 7:417–443, 2018.

[55] C. Spearman. The proof and measurement of association between two things. *American Journal of Psychology*, 1904.

[56] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. In *Workshop Proceedings of the International Conference on Learning Representations (ICLR)*, 2014.

[57] B. K. Sriperumbudur, A. Gretton, K. Fukumizu, B. Schölkopf, and G. R. Lanckriet. Hilbert space embeddings and metrics on probability measures. *J. Mach. Learn. Res.*, 11:1517–1561, aug 2010.

[58] M. Sundararajan, A. Taly, and Q. Yan. Axiomatic attribution for deep networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2017.

[59] M. Tan and Q. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *Proceedings of the International Conference on Machine Learning (ICML)*. PMLR, 2019.

[60] R. Tomsett, D. Harborne, S. Chakraborty, P. Gurram, and A. Preece. Sanity checks for saliency metrics. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2019.

[61] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *Proceedings of the IEEE European Conference on Computer Vision (ECCV)*, 2014.

[62] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

A On the Orthogonal Decomposition Property 1

In this part, we state the orthogonal decomposition Property, motivate its importance with a pedagogical example, and finally prove Proposition 1, which enables the decomposition property in the context of HSIC attribution method.

A.1 Orthogonal Decomposition Property

Let $\mathbf{x} = \{\mathrm{x}_1, ..., \mathrm{x}_n\} \in \mathcal{X}^n$ be a set of n univariate random input variables. For any subset $A = \{l_1, ..., l_{|A|}\} \subseteq \{1, ..., n\}$, we denote $\mathbf{x}_A = (\mathrm{x}_{l_1}, ..., \mathrm{x}_{l_{|A|}})$ the vector of input variables with indices in A. Let $\mathbf{y}$ the random output variable defined by $\mathbf{y} = \boldsymbol{f}(\mathbf{x})$, $\mathcal{F}$ the RKHS defined by the kernel $k_A : \mathcal{X}^{|A|} \to \mathbb{R}$ and $\mathcal{G}$ the RKHS defined by the kernel $l : \mathcal{Y} \to \mathbb{R}$.

In [11], the author shows that for any choice of kernel l, if we respect some constraints on the kernel k_A, we can construct indices $HSIC(\mathbf{x}_A, \mathbf{y})$ that satisfy the following decomposition property.

Property 2 (Decomposition property)**.** *For any kernel l, the kernel k_A satisfies the decomposition property if:*

$$HSIC(\mathbf{x}, \mathbf{y}) = \sum_{A \subseteq \{1,...,n\}} HSIC_A, \tag{7}$$

where each term $HSIC_A$ is given by

$$HSIC_A = \sum_{B \subseteq A} (-1)^{|A|-|B|} HSIC(\mathbf{x}_B, \mathbf{y}),$$

and $HSIC(\mathbf{x}_B, \mathbf{y})$ is defined as in equation (1) with kernels l and k_A.

The constraints on the kernel k_A are detailed in the main document and in the last section of this appendix. Before describing these constraints and how to fulfill them with Proposition 1, let us illustrate the importance of the property with a motivating, pedagogical example.

A.2 Motivating example

In this section, we introduce a pedagogical example to motivate the interest in assessing the interactions and the importance of the Orthogonal Decomposition Property in that regard. Let $f : [0, 2]^3 \to \{0, 1\}$ such that

$$\mathrm{y} = f(\mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_3) = \begin{cases} 1 & \text{if } \mathrm{x}_1 \in [0,1], \mathrm{x}_2 \in [1,2], \mathrm{x}_3 \in [0,1], \\ 1 & \text{if } \mathrm{x}_1 \in [0,1], \mathrm{x}_2 \in [0,1], \mathrm{x}_3 \in [1,2], \\ 0 & \text{otherwise.} \end{cases}$$

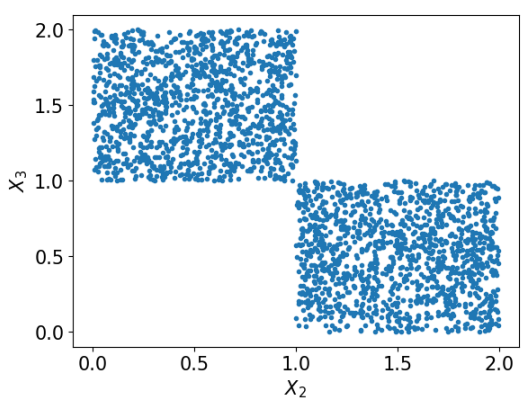

Figure 1: Input points $X_i = (\mathrm{x}_{1,i}, \mathrm{x}_{2,i}, \mathrm{x}_{3,i})$ for which $f(X_i) = 1$ with respect to x_2 and x_3.

The relation between x_2, x_3 and the output of function f is illustrated on Figure 7. Here x_i is analogous to M_i. In that case, it is clear that x_1 is important to explain the output. However, assessing

the effect of x_2 and x_3 is more difficult. Given the definition of f; they are important to explain the output y, but it can be shown theoretically that $HSIC(x_2, y) = 0$ and $HSIC(x_3, y) = 0$ [34]. This motivates to assess the interactions between input variables. One way to retrieve the information that x_2 and x_3 are important is to assess $HSIC(x_{2,3}, y)$, where $x_{2,3} = (x_2, x_3)$

One could assess $HSIC(x_{2,3}, y)$ without any constraints on the kernel k, and the obtained value for $HSIC(x_{2,3}, y)$ would indeed be > 0 . However, by doing so, we would also obtain that $HSIC(x_{1,2}, y) > 0$ and $HSIC(x_{1,3}, y) > 0$, whereas x_1 does not interact with x_2 and x_3, only because of the individual effect of x_1. We empirically illustrate this by assessing these metrics using the estimator of Eq. 2 with $p = 10000$, and kernels k, l chosen as the Radial Basis Function (RBF). The results are found in Table 1 below and show that:

- $HSIC(x_2, y) \approx HSIC(x_3, y) \approx 0$
- $HSIC(x_{1,2}, y) \approx HSIC(x_{1,3}, y) > HSIC(x_{2,3}, y)$

$HSIC(x_1, y)$	$HSIC(x_2, y)$	$HSIC(x_3, y)$	$HSIC(x_{1,3}, y)$	$HSIC(x_{1,2}, y)$	$HSIC(x_{2,3}, y)$
1.79×10^{-2}	2.28×10^{-6}	9.63×10^{-6}	1.36×10^{-2}	1.36×10^{-2}	2.92×10^{-3}

Table 1: HSIC metrics with k taken as RBF

In order to correctly assess the pairwise interactions of input variables x_1 and x_2, one has to remove the individual effect of each variable from the $HSIC(x_{1,2}, y)$. The orthogonal decomposition property [11] allows to do so by simply computing $HSIC_{inter}(x_{1,2}, y)$ as

$$HSIC_{inter}(x_{1,2}, y) = HSIC(x_{1,2}, y) - HSIC(x_1, y) - HSIC(x_2, y)$$

If the decomposition property does not hold, we are not guaranteed to fully remove the individual effect of x_1 and x_2 using the previous formula. We estimate $HSIC_{inter}(x_{1,2}, y)$ when the kernel k satisfies the decomposition property (in that case we choose a Sobolev kernel as in [1]), and when it does not, and show that the correct information of $HSIC_{inter}(x_{1,2}, y)$ is only retrieved when the decomposition property is satisfied. As previously, this is illustrated in the experiment, whose results are found in Table 2.

	$HSIC(x_2, y)$	$HSIC(x_3, y)$	$HSIC(x_{1,3}, y)$
k Sobolev	7.68×10^{-6}	2.83×10^{-6}	7.85×10^{-4}
k RBF	-4.35×10^{-3}	-4.30×10^{-3}	2.91×10^{-3}

Table 2: HSIC metrics for assessing interactions, when k satisfies (Sobolev) / does not satisfy (RBF) the orthogonal decomposition property

In that case, with k satisfying the orthogonal decomposition property (Sobolev), we retrieve that $HSIC_{inter}(x_{1,2}, y) \approx HSIC_{inter}(x_{1,3}, y) \approx 0$ and $HSIC_{inter}(x_{2,3}, y)$ is significant. When k does not satisfy the property (RBF), the values are not relevant (a negative value has no meaning since the metric is a distance)

A.3 Proof of Proposition 1

To benefit from Property 2, the kernel k_A must satisfy the following assumption [11]:

Assumption 1. *The kernel k_A satisfies Property 2 if*

$$k_A(\boldsymbol{x}_A, \boldsymbol{x}'_A) = \prod_{i \in A} (1 + k_0(x_i, x'_i)),$$

where

$$k_0(x, x') = k(x, x') - \frac{\int k(x,t)dP(t) \int k(x',t)dP(t)}{\int \int k(s,t)dP(s)dP(t)}.$$

We now recall and prove the introduced Proposition 1 defined in Section 3.3.

Proposition 1. *Let* x *a Bernoulli variable of parameter* $p = 1/2$*, and* $\delta(x = x')$ *the dirac kernel such that* $\delta(x = x') = 1$ *if* $x = x'$ *and* 0 *otherwise. Let* k_0 *be defined as in equation (4). Then, the kernel* k_A *satisfies the decomposition property (Property 1) if it is defined according to Assumption 1, with*

$$k_0(x, x') = \delta(x = x') - \frac{1}{2}. \tag{8}$$

Proof. Let s and t be two iid random Bernoulli variables of parameter p with probability density functions p_{s} and p_{t}. We have that

$$\begin{cases} dP(s) = p_{\mathrm{s}}(s)ds = \big(p\delta(s = 1) + (1 - p)\delta(s = 0)\big)ds \\ dP(t) = p_{\mathrm{t}}(t)dt = \big(p\delta(t = 1) + (1 - p)\delta(t = 0)\big)dt. \end{cases}$$

Now, let's consider two Bernoulli variables x and x′, two samples $x \sim$ x and $x' \sim$ x′, and a kernel k such that $k(x, x') = \delta(x = x')$.

- if x $\neq$ x′

$$\begin{cases} \int k(x,t)dP(t) \int k(x,s)dP(s) = p(1-p) \\ \int \int k(s,t)dP(s)dP(t) = p^2 + (1-p)^2 \end{cases}$$

- if x = x′ = 0

$$\begin{cases} \int k(x,t)dP(t) \int k(x,s)dP(s) = p^2 \\ \int \int k(s,t)dP(s)dP(t) = p^2 + (1-p)^2 \end{cases}$$

- if x = x′ = 1

$$\begin{cases} \int k(x,t)dP(t) \int k(x,s)dP(s) = (p-1)^2 \\ \int \int k(s,t)dP(s)dP(t) = p^2 + (1-p)^2 \end{cases}$$

Therefore, since $p = \frac{1}{2}$,

$$\frac{\int k(x,t)dP(t) \int k(x',t)dP(t)}{\int \int k(s,t)dP(s)dP(t)} = \frac{1}{2},$$

so the kernel

$$k_0(x, x') = \delta(x = x') - \frac{1}{2}$$

satisfies the decomposition property 2. □

B Complete fidelity results

	Method	*ResNet50*	*VGG16*	*EfficientNet*	*MobileNetV2*	Exec. time (s)
Del. (↓)						
White-box	Saliency [48]	0.158 ± 0.006	0.120 ± 0.005	0.091 ± 0.003	0.113 ± 0.004	0.360
	Grad.-Input [47]	0.153 ± 0.006	0.116 ± 0.004	0.084 ± 0.003	0.110 ± 0.004	0.023
	Integ.-Grad. [58]	0.138 ± 0.005	0.114 ± 0.004	**0.078** ± 0.002	0.096 ± 0.004	**1.024**
	SmoothGrad [50]	0.127 ± 0.005	0.128 ± 0.005	0.094 ± 0.003	**0.088** ± 0.003	0.063
	GradCAM++ [45]	**0.124** ± 0.004	**0.105** ± 0.003	0.112 ± 0.005	0.106 ± 0.005	0.127
	VarGrad [45]	0.134 ± 0.005	0.229 ± 0.007	0.224 ± 0.007	0.097 ± 0.004	0.097
Black-box	LIME [41]	0.186 ± 0.006	0.258 ± 0.008	0.186 ± 0.007	0.148 ± 0.006	6.480
	Kernel Shap [32]	0.185 ± 0.006	0.165 ± 0.006	0.164 ± 0.006	0.149 ± 0.006	4.097
	RISE [36]	0.114 ± 0.004	0.106 ± 0.004	0.113± 0.005	0.115 ± 0.004	8.427
	Sobol [13]	0.121 ± 0.003	0.109 ± 0.004	0.104 ± 0.003	0.107 ± 0.004	5.254
	$\mathcal{H}_i^p$ eff. (ours)	**0.106** ± 0.003	**0.100** ± 0.004	**0.095** ± 0.003	**0.094** ± 0.003	**0.956**
	$\mathcal{H}_i^p$ acc. (ours)	**0.105** ± 0.003	**0.099** ± 0.004	**0.094** ± 0.003	**0.093** ± 0.003	1.668
Ins. (↑)						
White-box	Saliency [48]	0.357 ± 0.009	0.286 ± 0.009	0.224 ± 0.008	0.246 ± 0.008	0.360
	Grad.-Input [47]	0.363 ± 0.010	0.272 ± 0.008	0.220 ± 0.009	0.231 ± 0.007	0.023
	Integ.-Grad. [58]	0.386 ± 0.010	0.276 ± 0.009	0.248 ± 0.008	0.258 ± 0.008	**1.024**
	SmoothGrad [50]	0.379 ± 0.010	0.229 ± 0.008	0.172 ± 0.006	0.246 ± 0.008	0.063
	GradCAM++ [45]	0.497 ± 0.010	**0.413** ± 0.010	**0.316** ± 0.009	0.387 ± 0.009	0.127
	VarGrad [45]	**0.527** ± 0.010	0.241 ± 0.008	0.222 ± 0.007	**0.399** ± 0.009	0.097
Black-box	LIME [41]	0.472 ± 0.010	0.273 ± 0.009	0.223 ± 0.007	0.384 ± 0.009	6.480
	Kernel Shap [32]	0.480 ± 0.010	0.393 ± 0.009	0.367 ± 0.008	0.383 ± 0.009	4.097
	RISE [36]	**0.554** ± 0.010	**0.485** ± 0.010	**0.439** ± 0.009	**0.443** ± 0.009	8.427
	Sobol [13]	0.370 ± 0.009	0.313 ± 0.009	0.309 ± 0.009	0.331 ± 0.009	5.254
	$\mathcal{H}_i^p$ eff. (ours)	0.470 ± 0.011	0.387 ± 0.010	0.357 ± 0.009	0.381 ± 0.009	**0.956**
	$\mathcal{H}_i^p$ acc. (ours)	0.481 ± 0.011	0.395 ± 0.011	0.366 ± 0.009	0.392 ± 0.009	1.668

Table 3: **Deletion** and **Insertion** scores obtained on 1,000 ImageNet validation set images (For Deletion, lower is better and for Insertion, higher is better). The execution times are averaged over 100 explanations of ResNet50 predictions with a RTX Quadro 8000 GPU. The first and second best results are **bolded** and underlined.

	Method	*ResNet50*	*VGG16*	*EfficientNet*	*MobileNetV2*	Exec. time (s)
White-box	Saliency [48]	0.192 ± 0.034	0.092 ± 0.035	0.102 ± 0.029	**0.172** ± 0.030	0.360
	Grad.-Input [47]	0.157 ± 0.034	0.066 ± 0.029	0.085 ± 0.030	0.116 ± 0.029	0.023
	Integ.-Grad. [58]	0.162 ± 0.033	0.073 ± 0.029	**0.139** ± 0.028	**0.157** ± 0.030	**1.024**
	SmoothGrad [50]	**0.230** ± 0.032	0.087 ± 0.030	**0.101** ± 0.030	0.126 ± 0.028	0.063
	GradCAM++ [45]	0.142 ± 0.032	**0.143** ± 0.032	**0.128** ± 0.031	0.131 ± 0.029	0.127
	VarGrad [45]	0.021 ± 0.022	0.022 ± 0.020	0.001 ± 0.003	0.101 ± 0.032	0.097
Black-box	LIME [41]	0.110 ± 0.033	0.015 ± 0.032	0.000 ± 0.024	0.055 ± 0.031	6.480
	Kernel Shap [32]	0.104 ± 0.033	0.068 ± 0.034	0.079 ± 0.032	0.051 ± 0.031	4.097
	RISE [36]	0.182 ± 0.034	0.099 ± 0.034	**0.133** ± 0.036	**0.123** ± 0.031	8.427
	Sobol [13]	**0.230** ± 0.034	**0.110** ± 0.030	**0.141** ± 0.034	**0.131** ± 0.030	5.254
	$\mathcal{H}_i^p$ eff. (ours)	**0.202** ± 0.034	**0.116** ± 0.034	**0.154** ± 0.035	**0.111** ± 0.031	**0.956**
	$\mathcal{H}_i^p$ acc. (ours)	0.187 ± 0.035	**0.136** ± 0.030	**0.155** ± 0.035	**0.120** ± 0.031	1.668

Table 4: μ**Fidelity** scores, obtained on 1,000 images from ImageNet validation set. Higher is better. The first and second best results are **bolded** and underlined. The execution times are averaged over 100 explanations of ResNet50 predictions with a RTX Quadro 8000 GPU.

C Additional visualizations on object detection explanations

C.1 Visualizations

In this part we provide a sample of visualizations of object detection explanations for HSIC, RISE and KernelShap. HSIC seems more robust than the two other methods that are often blurry and sometimes fail. These images are taken from the 40 first images of COCO dataset. Out of transparancy, we

provide all the 40 first explanations in the github repository found at `https://anonymous.4open.science/r/HSIC-Attribution-Method-C684`.

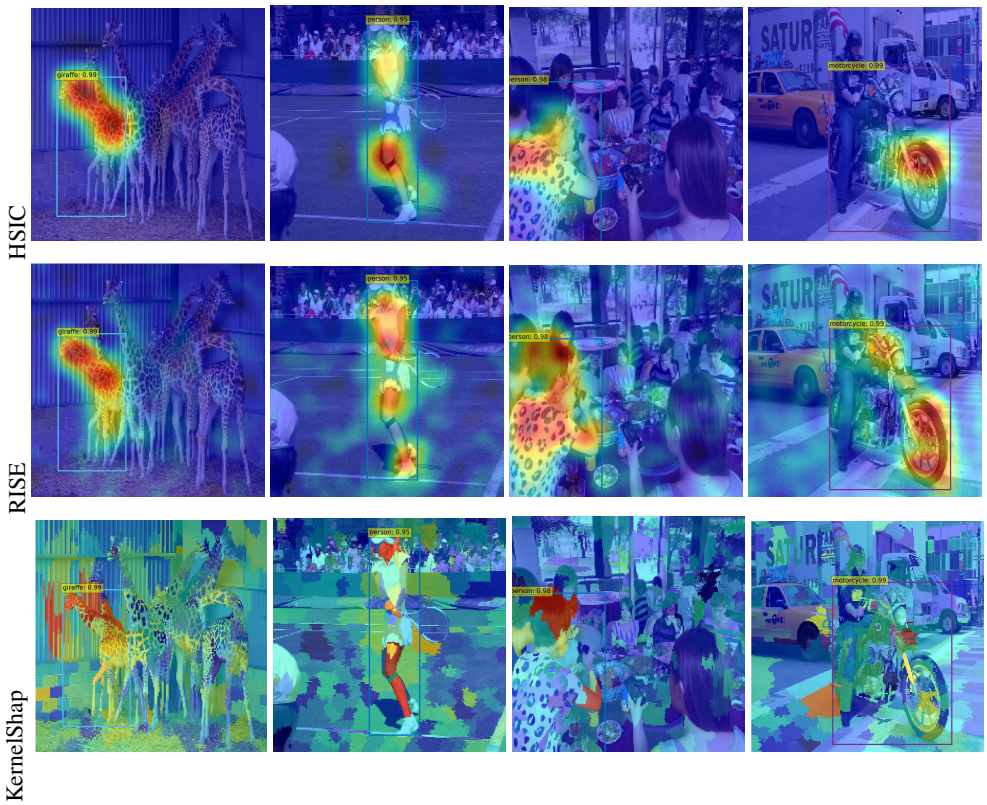

Figure 2: Visualisations of object detection explanations (1/2).

C.2 Error explanations oh HSIC against RISE

In this section, we show explanations of RISE and KernelShap for the image where Yolov4 erroneously recognizes a cat instead of a zebra. HSIC manages to find an explanation for this error while both RISE and KernelShap fail, even for different grid sizes.

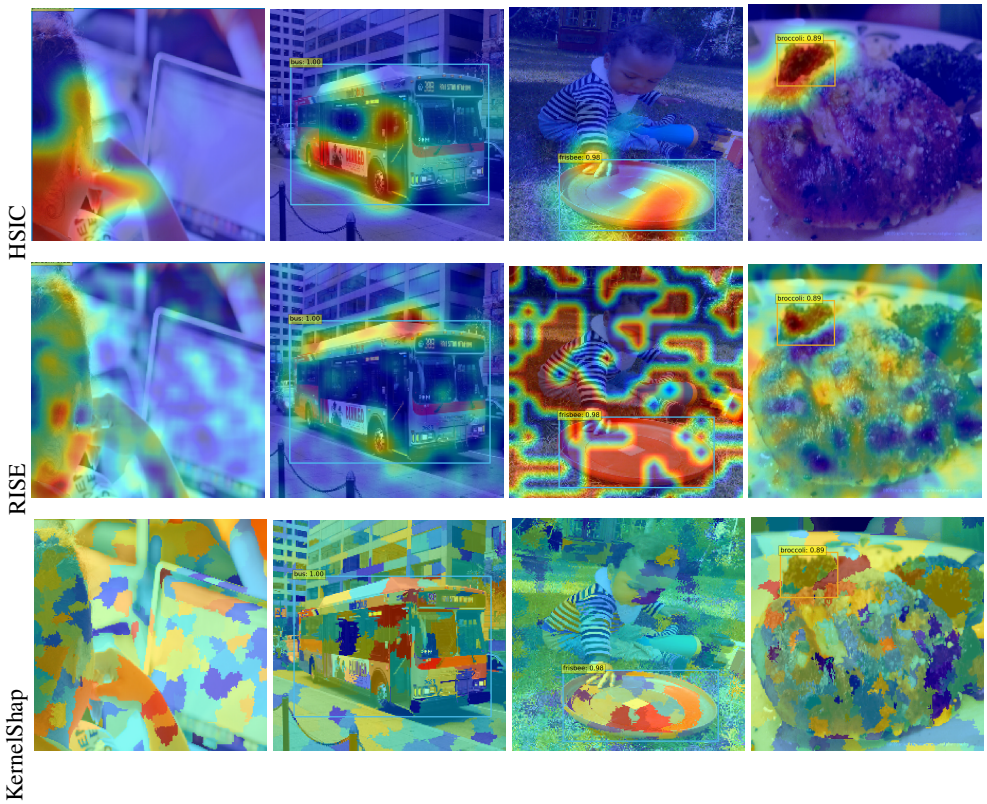

Figure 3: Visualisations of object detection explanations (2/2).

D Additional visualizations of HSIC attribution method on ImageNet

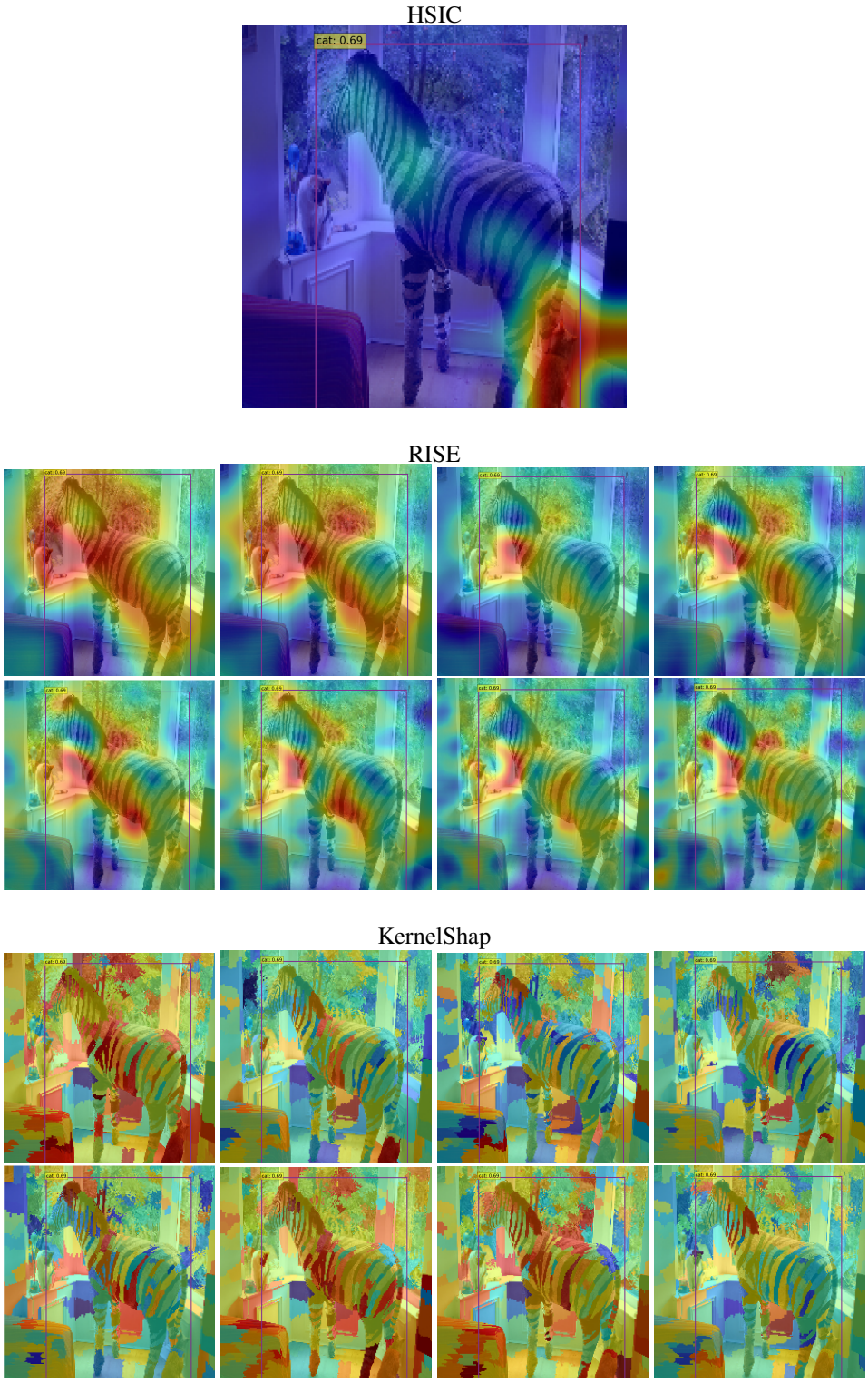

Figure 4: Visualizations of object detection explanations for a model error with HSIC method. Blurry explanations for different grid sizes with RISE and KernelShap.

E Attribution methods

In the following section, we give the formulation of the different attribution methods used in this work. The library used to generate the attribution maps is Xplique [15]. By simplification of notation,

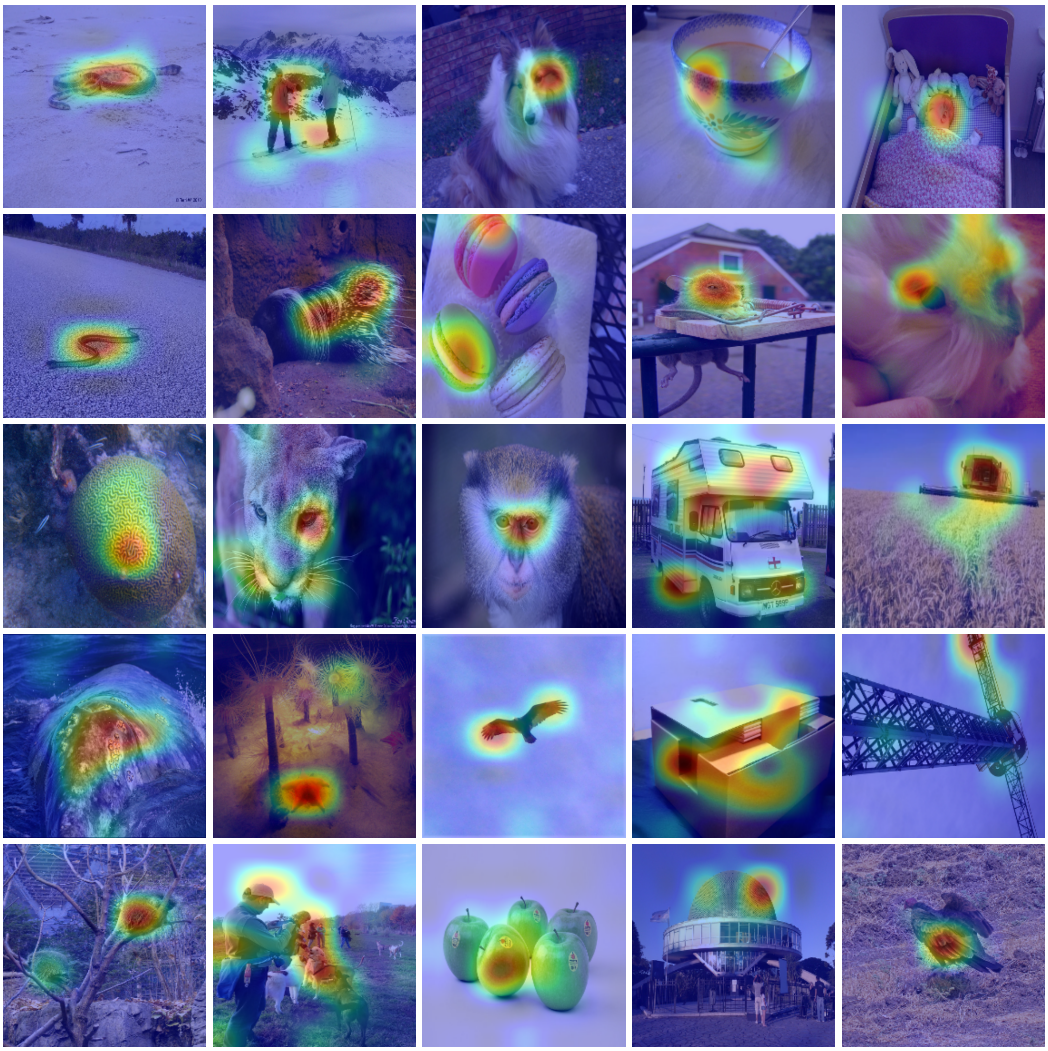

Figure 5: Explanations for ImageNet with HSIC eff.

we define $\boldsymbol{f}(x)$ the logit score (before softmax) for the class of interest (we omit c). We recall that an attribution method provides an importance score for each input variable x_i. We will denote the explanation functionnal mapping an input of interest $x = (x_1, ..., x_d)$ as $\boldsymbol{g}(x)$.

Saliency [48] is a visualization technique based on the gradient of a class score relative to the input, indicating in an infinitesimal neighborhood which pixels must be modified to most affect the score of the class of interest.

$$\boldsymbol{g}(x) = ||\nabla_x \boldsymbol{f}(x)||$$

Gradient $\odot$ Input [46] is based on the gradient of a class score relative to the input, element-wise with the input, it was introduced to improve the sharpness of the attribution maps. A theoretical analysis conducted by [3] showed that Gradient $\odot$ Input is equivalent to ϵ-LRP and DeepLIFT [46] methods under certain conditions – using a baseline of zero and with all biases to zero.

$$\boldsymbol{g}(x) = x \odot ||\nabla_x \boldsymbol{f}(x)||$$

Integrated Gradients [58] consists of summing the gradient values along the path from a baseline state to the current value. The baseline x_0 used is zero. This integral can be approximated with a set

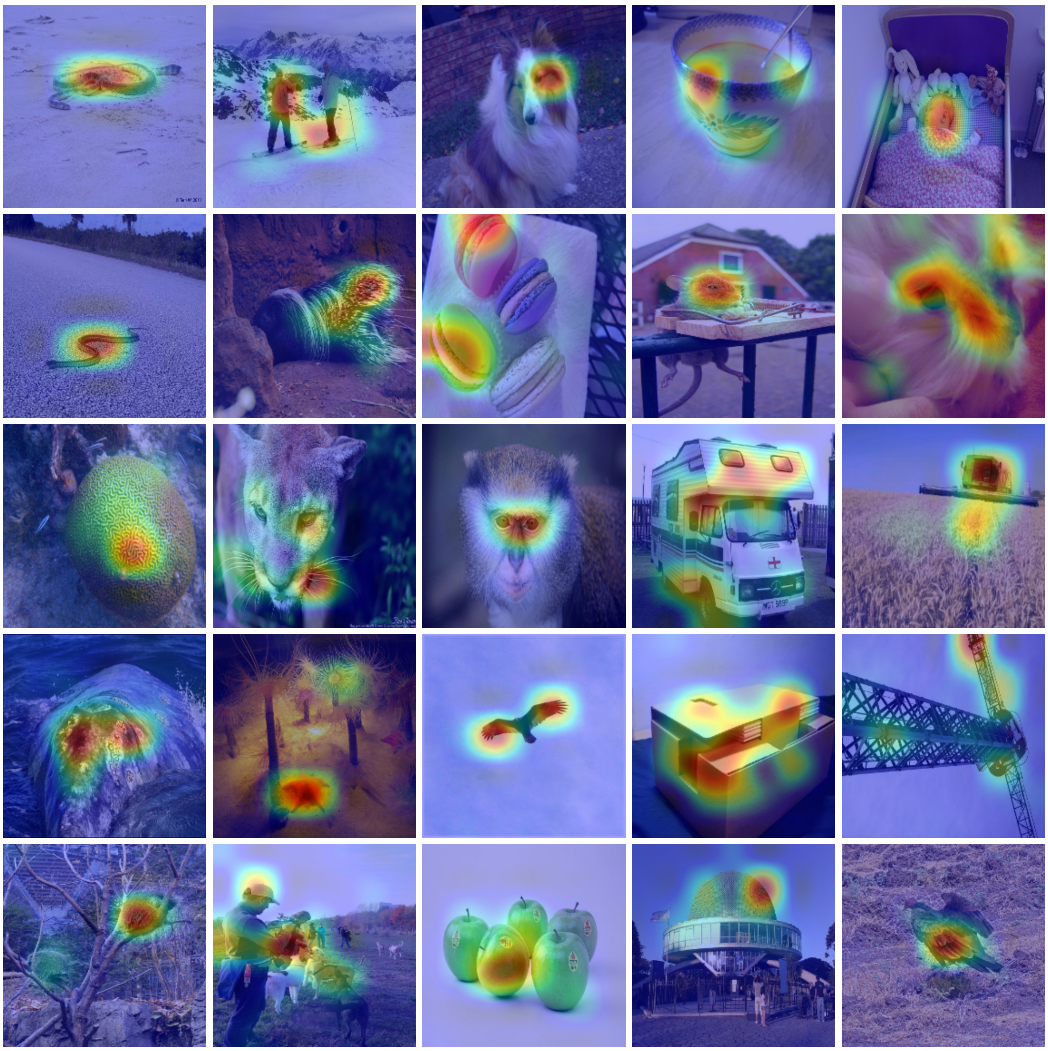

Figure 6: Explanations for ImageNet with HSIC acc.

of m points at regular intervals between the baseline and the point of interest. In order to approximate from a finite number of steps, we use a Trapezoidal rule and not a left-Riemann summation, which allows for more accurate results and improved performance (see [53] for a comparison).

$$\boldsymbol{g}(x) = (x - x_0) \int_0^1 \nabla_x \boldsymbol{f}(x_0 + \alpha(x - x_0)))d\alpha$$

SmoothGrad [50] is also a gradient-based explanation method, which, as the name suggests, averages the gradient at several points corresponding to small perturbations (drawn i.i.d from an isotropic normal distribution of standard deviation σ) around the point of interest. The smoothing effect induced by the average help reducing the visual noise, and hence improve the explanations. The attribution is obtained by averaging after sampling m points. For all the experiments, we took $m = 80$ and $\sigma = 0.2 \times (x_{\max} - x_{\min})$ where $(x_{\min}, x_{\max})$ being the input range of the dataset.

$$\boldsymbol{g}(x) = \mathop{\mathbb{E}}_{\delta \sim \mathcal{N}(0, \boldsymbol{I}\sigma)} (\nabla_x \boldsymbol{f}(x + \delta))$$

VarGrad [26] is similar to SmoothGrad as it employs the same methodology to construct the attribution maps: using a set of m noisy inputs, it aggregate the gradients using the variance rather than the mean. For the experiment, m and σ are the same as Smoothgrad. Formally:

$$\boldsymbol{g}(x) = \underset{\delta \sim \mathcal{N}(0, \boldsymbol{I}\sigma)}{\mathbb{V}} (\nabla_x \boldsymbol{f}(x + \delta))$$

Grad-CAM [45] can only be used on Convolutional Neural Network (CNN). Thus we couldn't use it for the MNIST dataset. The method uses the gradient and the feature maps $\boldsymbol{A}^k$ of the last convolution layer. More precisely, to obtain the localization map for a class, we need to compute the weights α_c^k associated to each of the feature map activation $\boldsymbol{A}^k$, with k the number of filters and Z the number of features in each feature map, with $\alpha_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\partial \boldsymbol{f}(x)}{\partial \boldsymbol{A}_{ij}^k}$ and

$$\boldsymbol{g} = \max(0, \sum_k \alpha_k^c \boldsymbol{A}^k)$$

As the size of the explanation depends on the size (width, height) of the last feature map, a bilinear interpolation is performed in order to find the same dimensions as the input. For all the experiments, we used the last convolutional layer of each model to compute the explanation.

Grad-CAM++ (G+) [7] is an extension of Grad-CAM combining the positive partial derivatives of feature maps of a convolutional layer with a weighted special class score. The weights $\alpha_c^{(k)}$ associated with each feature map are computed as follows:

$$\alpha_k^c = \sum_i \sum_j [\frac{\frac{\partial^2 \boldsymbol{f}(x)}{(\partial \boldsymbol{A}_{ij}^{(k)})^2}}{2 \frac{\partial^2 \boldsymbol{f}(x)}{(\partial \boldsymbol{A}_{ij}^{(k)})^2} + \sum_i \sum_j \boldsymbol{A}_{ij}^{(k)} \frac{\partial^3 \boldsymbol{f}(x)}{(\partial \boldsymbol{A}_{ij}^{(k)})^3}}]$$

Occlusion [61] is a sensitivity method that sweeps a patch that occludes pixels over the images using a baseline state and uses the variations of the model prediction to deduce critical areas. For all the experiments, we took a patch size and a patch stride of $\frac{1}{7}$ of the image size. Moreover, the baseline state x_0 was zero.

$$\boldsymbol{g}(x)_i = \boldsymbol{f}(x) - \boldsymbol{f}(x_{[x_i=0]})$$

RISE [36] is a black-box method that consists of probing the model with N randomly masked versions of the input image to deduce the importance of each pixel using the corresponding outputs. The masks $\boldsymbol{m} \sim \mathcal{M}$ are generated randomly in a subspace of the input space. For all the experiments, we use a subspace of size 7×7 and $\mathbb{E}(\mathcal{M}) = 0.5$.

$$\boldsymbol{g}(x) = \frac{1}{\mathbb{E}(\mathcal{M})N} \sum_{i=0}^{N} \boldsymbol{f}(x \odot \boldsymbol{m}_i) \boldsymbol{m}_i$$

F Evaluation

For the purpose of the experiments, three fidelity metrics have been chosen. For the whole set of metrics, $\boldsymbol{f}(x)$ score is the score after softmax of the models. We first describe these metrics and then discuss the trade-off between Deletion and Insertion scores.

F.1 Definitions

Deletion. [36] The first metric is Deletion, it consists in measuring the drop in the score when the important variables are set to a baseline state. Intuitively, a sharper drop indicates that the explanation method has well identified the important variables for the decision. The operation is repeated on the whole image until all the pixels are at a baseline state. Formally, at step k, with $\boldsymbol{u}$ the most important variables according to an attribution method, the Deletion$^{(k)}$ score is given by:

$$\text{Deletion}^{(k)} = \boldsymbol{f}(x_{[x_{\boldsymbol{u}}=x_0]})$$

We then measure the AUC of the Deletion scores. For all the experiments, the baseline state is fixed at $x_0 = 0$.

Insertion. [36] Insertion consists in performing the inverse of Deletion, starting with an image in a baseline state and then progressively adding the most important variables. Formally, at step k, with $\boldsymbol{u}$ the most important variables according to an attribution method, the Insertion$^{(k)}$ score is given by:

$$\text{Insertion}^{(k)} = \boldsymbol{f}(x_{[x_{\overline{\boldsymbol{u}}}=x_0]})$$

We then measure the AUC of the Deletion scores. The baseline is the same as for Deletion.

μ**Fidelity** [5] consists in measuring the correlation between the fall of the score when variables are put at a baseline state and the importance of these variables. Formally:

$$\mu\text{Fidelity} = \underset{\substack{\boldsymbol{u}\subseteq\{1,\dots,d\}\\ |\boldsymbol{u}|=k}}{\text{Corr}} \left(\sum_{i\in\boldsymbol{u}} \boldsymbol{g}(x)_i, \boldsymbol{f}(x) - \boldsymbol{f}(x_{[x_{\boldsymbol{u}}=x_0]}) \right)$$

For all experiments, k is equal to 20% of the total number of variables and the baseline is the same as the one used by Deletion.

F.2 Trade-off between Insertion and Deletion

Deletion and Insertion metrics consist in measuring AUC of scores that respectively decrease and increase when deleting and adding patches, starting from a baseline image. Since the patches deleted/added are those that are the most important (in the sense of the tested attribution method), most of the score will come from the first patch deletions/additions. Using those different methods has two important consequences, detailed below.

Deletion is preferable There is a key difference between those two evaluations that makes Deletion more suited to explanation evaluation than Insertion. In Deletion, since we start from the original image and sequentially delete patches, the score is tested in a region of the input image space that is close to the input image. On the contrary, Insertion starts from an arbitrary baseline (here, pure black image), which is far from the input image. It is likely that the value of the baseline has an undesired impact on the score for Insertion. That is why we tend to favor Deletion over Insertion.

Some methods are more suited to Deletion or Insertion Since Deletion measures a drop in the score, the faster the score drops, the better the metric. Hence, Deletion will favor methods that sharply identify important regions. On the contrary, since Insertion starts from an arbitrary baseline image, if the explanation map is more spread out, more relevant secondary information will be added, so the score will be better. To illustrate this observation, in table 5 we show the value of Insertion and Deletion metrics for HSIC method and for different grid sizes, obtained after a grid search for MobileNetV2 on 1000 ImageNet validation images. The metrics are averaged over 27 runs (with a different number of samples and different samplers). Table 5 gives an idea of the trend of the evolution of Insertion and Deletion with respect to the grid size. As we can see, Deletion improves when the grid size increases, i.e. when the explanation map becomes sharper, and Insertion improves when the grid size decreases, i.e. when the map becomes more spread out.

grid size	5	6	7	8	9	10
Insertion $\times 10^{-1}$	4.14	4.02	3.90	3.72	3.54	3.40
Deletion $\times 10^{-1}$	1.01	0.97	0.94	0.93	0.92	0.90

Table 5: Result of a grid search for MobileNetV2

This trend also explains why RISE shines in the Insertion benchmark and why our HSIC attribution method dominates the Deletion benchmark. Indeed, as we can see in the maps of Appendix C, RISE saliency maps are way more spread out than HSIC's, which are sharper.

G Additional experiments on stability

In this section, we report the evolution of the Deletion score for HSIC, RISE, and Sobol with respect to the number of forward passes, with a Resnet50 on 100 Imagenet validation images.

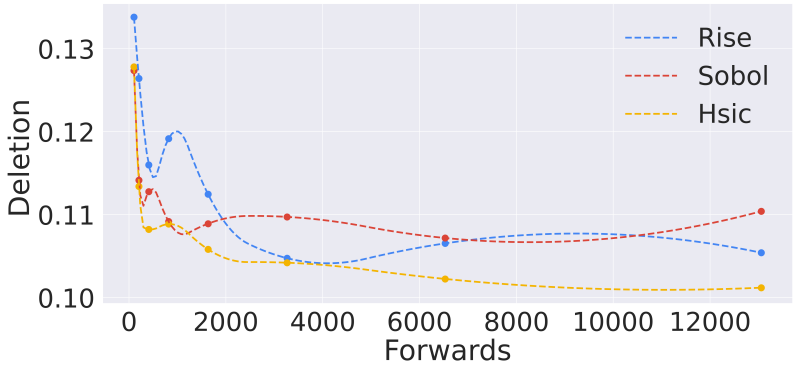

Figure 7: Deletion score for HSIC, RISE, and Sobol with respect to the number of forward passes

The scores for Sobol and RISE are less stable than for HSIC, which corroborates that HSIC attribution method can be used with fewer forward passes.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] The limitations stem from the complexity of computing HSIC mentioned in Sections 3.2 and 3.4. We provided a vectorized implementation to alleviate this limitation.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]] There is no negative societal impact specific to this work that is not shared with common XAI techniques.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] In Appendix A
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] URL is found in page 1.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Error bars have been computed but left in the Appendix to make the presentation of Tables 1 and 2 lighter. Nonetheless, they are taken into account in the tables and in the comments to assess the statistical significance of results
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]