Image and Vision Computing 124 (2022) 104516

Contents lists available at ScienceDirect

Image and Vision Computing

journal homepage: www.elsevier.com/locate/imavis

Output-targeted baseline for neuron attribution calculation

Rui Shi [a], Tianxing Li [a], Yasushi Yamaguchi [b,*]

[a] Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China
[b] Department of General Systems Studies, The University of Tokyo, Tokyo 153-8902, Japan

ARTICLE INFO

Article history:
Received 26 May 2022
Accepted 28 June 2022
Available online 01 July 2022

Keywords:
Convolutional neural networks
Network interpretability
Attribution methods
Shapley values

ABSTRACT

Attribution methods explaining a particular decision for a given convolutional neural network (CNN) have gained a lot of attention over the last few years. Among them, approximation methods of Shapley values are considered to be better ways of assigning attribution scores such that several desirable axioms are satisfied. Nevertheless, these attribution scores may still be misleading or inaccurate due to the inappropriate selection of a baseline which is necessary to apply Shapley values to CNNs. Previous baseline studies have focused on developing a generic baseline selection method for all approximation methods; however, we find that designing a baseline under the essence of the selected approximation method itself produces better results than generic ones. With this observation, we propose two primal baseline properties for Aumann–Shapley-based attributions and design a general objective function of generating a baseline iteratively by gradient descent. To increase efficiency, we further reduce the objective function into a quadratic optimization problem where the gradients only need to be calculated once. We show that our method produces better attribution results than several state-of-the-art baseline selections and attribution methods on both qualitative and quantitative experiments.

1. Introduction

Inducing a convolutional neural network (CNN) to learn important features from a large-scale image dataset is a powerful paradigm, yet this introduces a challenging interpretability problem of explaining complex network behaviors [1–3]. In the realm of interpretability, attribution methods aiming at explaining a particular output by assigning a scalar attribution score to each feature of a CNN have drawn an increasing attention over the past few years [4]. Specifically, for a particular output indexed with c, the goal of attribution methods is to compute attribution scores $\mathbf{R}^c \in \mathbb{R}^{W\times H\times K}$ in a hidden layer, where W,H,K are width, height, and channel number of feature maps in a hidden layer.

To further improve the reliability of attribution explanations in theoretical, some researchers endorsed an axiomatic approach [5–9]. Along with the axiomatic approach, the studies in game theory suggest Shapley values [10] as a unique way of assigning attribution scores such that desirable axioms (see Appendix A for details) can be satisfied [8]. The original Shapley method calculates the *averaged marginal contribution* of a feature under all possible feature coalitions to generate its attribution score. Unfortunately, computing the exact Shapley values is an NP-hard problem and only feasible for less than 20 features [5]. To compute Shapley values in polynomial time, many approximation methods have been proposed, *e.g.*, Aumann–Shapley (AS) method [11] and several expectation estimating methods. But, the methods estimating value function by conditional or interventional expectation may produce unreliable attributions [12]. Therefore, we select *Aumann–Shapley* method, which approximates Shapley values using the perfect sampling assumption, to calculate neuron attributions.

Although the idea of Aumann–Shapley method can be directly applied to calculate neuron attributions, the definition of marginal contributions requires a baseline $\overline{\mathbf{A}}$, also named "reference" [5], to simulate the missingness of features. Depending on the choice of baseline, the attribution results may change dramatically, because a Shapley-based explanation is essentially a function of difference between features and the baseline. Currently, Zero [9,13,14], Expectation [6], Blurring image [15], Uniform, Gaussian, and MaxDistance [16], and Neutral [17] baselines have been proposed as generic baseline selections. However, a generic baseline means that the essence of an approximation method itself cannot be taken into account. Specifically, Aumann–Shapley method inherently requires a baseline that only expresses the absence of features related to the particular output (see Sec. 4 for details). A baseline not targeted with the particular output will introduce misleading noise into Aumann–Shapley explanations. For example, attribution heatmaps may be disturbed by the neuron features irrelevant to the particular output, and therefore the target image region cannot be highlighted accurately.

* Corresponding author.
E-mail address: yama@g.ecc.u-tokyo.ac.jp (Y. Yamaguchi).

https://doi.org/10.1016/j.imavis.2022.104516
0262-8856/

In this context, we propose two baseline properties based on the essence of Aumann–Shapley method itself and demonstrate the significant influence of a baseline on calculating attributions. More specifically, this paper presents the following contributions.

1) We propose two primal baseline properties based on the definition of Aumann–Shapley values in CNNs, *i.e.*, attribution baseline ought to express the feature missingness of the particular output and ought to approach original features. We verify that baselines not satisfying these properties lead to misleading and degenerate attribution results through an experiment (Sec. 5.2)
2) Based on the characteristics of CNN neuron features in spatial and channel dimensions, we design two objective functions according to the baseline properties to generate *spatial* and *channel* masks by gradient descent, respectively. Then, we use the optimized masks to modify the original feature maps to generate the output-targeted baseline.
3) To decrease baseline calculation consumption, we propose to reduce the objective function into a quadratic optimization problem to calculate two masks with computing gradients only once. The faster calculating method is much more efficient in practice and also satisfies the baseline properties (Sec. 5.1). We further show that our method produces better results both qualitatively and quantitatively than several state-of-the-art baseline selections (Sec. 5.2) and attribution methods (Sec. 5.3).

2. Related work

2.1. Baseline selections

In practice, since the concept of feature missingness is domain-specific, the baseline definition is also varied to better represent the absence of information. Commonly, making the output score drop to zero or drop significantly is used to indicate feature missingness. For most domains, the zero baseline is reasonable to some degree [9,13,14]. Given a multilayer nonlinear network, without considering the influence of additive biases and all nonlinear activation functions, the zero baseline definitely results in a zero output. Although the output is often not zero because additive biases are usually not zero in practice, the zero baseline still can be considered as a canonical choice. Intuitively, when features are image pixels, a zero baseline is just a black or gray image that is able to represent the lack of features to humans. In image domain, generating baseline by blurring image is also widely illustrated [15,16]. However, for other domains, a baseline still needs to be selected with respect to the input instance at hand. For example, a zero baseline may not be able to represent the feature missingness in a hidden layer where the neuron value is a real number.

In addition to the zero baseline, several sophisticated attribution baseline selections have been proposed. Lundberg et al. [6] designed Expectation baseline that is calculated by averaging over randomly sampled feature coalition from the training set. However, sample unbalanced problem and the bias of the dataset itself can potentially lead to a serious decline in the performance of Expectation baseline. Their team further discussed effects of different types of baselines on CNNs and proposed Uniform noise baseline, Gaussian noise baseline, and MaxDistance baseline [16]. The MaxDistance baseline means find the farthest features in 2-norm but still in the original feature range. Recently, Izzo et al. [17] proposed Neutral baseline for multiple layer perceptrons (MLPs) by finding a point at which baseline values are determined by the network output being either above or below it. The major difference from other previous studies is that Neutral baseline does not attempt to reduce the output score to zero or significantly to represent feature missingness, but to find a decision boundary to express the absence of information. As all these considerations are derived with a clear objective, we believe that they are all reasonable for designing a baseline selection method.

With the recent research of taxonomy of baseline methods [18], baselines can be divided into two types, *i.e.*, static and dynamic baselines. Specifically, a baseline is static if it is the same for any input instance. Otherwise, a baseline is dynamic. Simply, a static baseline provides the same information for every observation, whereas a dynamic baseline provide different information. Thus, Zero [9,13,14], Expectation [6], and Neutral [17] baselines are static. Blurring image [15], Uniform, Gaussian, and MaxDistance [16] baselines are dynamic. Compared with static baselines, dynamic one is more flexible and easier to satisfy the requirements of Shapley approximation methods. On the other hand, all baseline selection methods mentioned above are trying to achieve a generic baseline; however, the essence of the Shapley approximation method is not taken into account. In fact, researchers recently expressed concerns about the appropriateness of baseline selection methods [18,19]. Therefore, the baseline selection should be implemented dynamically based on the essence of the attribution method itself.

2.2. Attribution methods

There has been a significant body of literature focusing on interpreting behaviors of deep neural networks. We focus on methods that can be applied to different network architectures and that are not restricted at the certain feature type. Although there are several criteria to categorize attribution methods, we discuss them according to whether they are based on Shapley values.

Simonyan et al. [20] utilized saliency information as the explanation of the particular output (Saliency). Shrikumar et al. [21] introduced an element-wise multiplication between the signed gradient and the input (GradxInput) as contributions. Smilkov et al. [22] added noise into original input and average the corresponding gradients to produce an explanation (SmoothGrad). Although vanilla gradients provide information about which features can be locally perturbed the least in sequence to maximally change the particular output, it does not help to compute the marginal contribution of a feature. To overcome this limitation, other methods are characterized by designing different backpropagation rules for different operations. Bach et al. [23] proposed Layer-wise Relevance Propagation (LRP) and Montavon et al. [7] built upon LRP to further propose deep Taylor decomposition. Shrikumar et al. [14,21] proposed Deep Learning Important Features (DeepLIFT) Rescale and RevealCancel. However, all above methods break at least one of the self-evident axioms as discussed in the studies [5,9].

Shapley values have been proved to satisfy several fundamental axioms [6,9,24]. Many approximation methods have been proposed to estimate Shapley values in polynomial time. Castro et al. [25] sampled a subset of all possible coalitions of features to approximate Shapley values. Ghorbani and Zou [26] sampled a subset of highly activated channels to improve sampling-based estimation methods. Although these random-sampling methods can handle the channel-level attribution calculation, we find they are still slow for neuron-level case, because the number of neurons is about hundreds of times that of channels. Sundararajan et al. [9] proposed Integrated Gradients (IntGrad) that can be regarded as computing Aumann–Shapley values using the zero baseline with some path modifications. Dhamdhere et al. [27] applied IntGrad to calculate neuron-level attributions. Although these attribution methods have better theoretical properties, they did not consider the significant influence of the baseline selection in CNNs. Chen et al. [28] proposed Deep Shapley Additive Explanations (DeepSHAP) to estimate Shapley values with a layer-wise chain rule; however, the generalization of the chain rule on Shapley values is not yet clear.

3. Neuron attributions

3.1. Aumann–Shapley values in CNNs

In game theory, Aumann–Shapley values are used to quantify the contribution of each player to the particular result when all players

participate in a game. The basic idea of Aumann–Shapley values can be extended to CNNs to compute neuron attributions. Consider a feed-forward CNN f that takes an image as input and produces feature maps $\mathbf{A} \in \mathbb{R}^{W\times H\times K}$ in the $\swarrow_{th}$ layer. The particular output $f^c(\mathbf{A})$ is generated when the feature maps are forwarded to the network and all previous layers are ignored. The original idea of Aumann–Shapley method is to evaluate marginal contribution on a "perfect" sample of the population of all neurons:

$$R^c_{i,j,k} = \int_{t=0}^{1} f^c\Big((1-t)\overline{\mathbf{A}}^c + t\mathbf{A} + \Delta A_{i,j,k}\Big) - f^c\Big((1-t)\overline{\mathbf{A}}^c + t\mathbf{A}\Big)dt, \tag{1}$$

where $(1-t)\overline{\mathbf{A}}^c + t\mathbf{A} + \Delta A_{i,j,k}$ can be considered as the coalition obtained after the neuron $A_{i,j,k}$ joins which can be regarded as the heuristic form of the diagonal formula. As t increases, the value increases diagonally from the baseline to the original feature maps. $\Delta\mathbf{A} = \mathbf{A} - \overline{\mathbf{A}}^c$, where $\overline{\mathbf{A}}^c$ is the baseline related to the particular output c. Note that, $\Delta A_{i,j,k}$ is not added to the tensor as a scalar, but is added to the corresponding value of the tensor. Then, this formula can be expanded by the Taylor series:

$$f^c\Big((1-t)\overline{\mathbf{A}}^c + t\mathbf{A} + \Delta A_{i,j,k}\Big) = f^c\Big((1-t)\overline{\mathbf{A}}^c + t\mathbf{A}\Big) + \Delta A_{i,j,k}\frac{\partial f^c\Big((1-t)\overline{\mathbf{A}}^c + t\mathbf{A}\Big)}{\partial A_{i,j,k}} + O\Big[\big(\Delta A_{i,j,k}\big)^2\Big], \tag{2}$$

where the remainder $O[(\Delta A_{i,\,j,\,k})^2]$ can be minimized when $\Delta A_{i,j,k}$ approaches 0. By substituting the above Taylor expansion into Eq. (1), the most common definition of Aumann–Shapley values can be obtained, *i.e.*,

$$R^c_{i,j,k} = \Delta A_{i,j,k}\int_{t=0}^{1}\frac{\partial f^c\Big((1-t)\overline{\mathbf{A}}^c + t\mathbf{A}\Big)}{\partial A_{i,j,k}}dt, \tag{3}$$

where Aumann–Shapley value $R^c_{i,\,j,\,k}$ is the gradient integral along this straight-line path. However, we cannot achieve the integral directly in neural networks because continuous gradients are not available. In addition, vectorized execution is necessary for efficiency, especially for tens of thousands of neurons in a hidden layer.

3.2. Neuron attribution calculation

To practically implement the attribution calculation in CNNs, we discretize the continuous integral using Gauss–Legendre quadrature and vectorize Eq. (3) as:

$$\mathbf{R}^c = \Delta\mathbf{A}\sum_{t=1}^{T}\frac{1}{\Big(1-\xi_t^2\Big)[P\prime_T(\xi_t)]^2}\frac{\partial f^c\Big(1/2\Big((1-\xi_t)\overline{\mathbf{A}}^c + (1+\xi_t)\mathbf{A}\Big)\Big)}{\partial\mathbf{A}}, \tag{4}$$

where T is the number of sample points in Gauss–Legendre quadrature for approximating the definite integral. ξ_t is the quadrature point of the T_{th} Legendre polynomial. $P_T\prime$ is the derivative of Legendre polynomials at the sample point. With a baseline and Eq. (4), we can calculate neuron attribution scores $\mathbf{R}^c$ by cumulating gradients at all points. Due to errors of the discretization, Eq. (4) prevents attribution scores from perfectly satisfying Axiom 1 (Appendix A). But we find that, for most cases, resulting attribution scores roughly add up to the output score under the number of sample points $T = 50$, *i.e.*, Axiom 1 can be approximately satisfied with the discrete calculation.

There are often hundreds of features along the channel which are difficult for humans to understand. To make attribution scores human scale, we could discard channel information and only retain the spatial distribution of features by generating an attribution intensity map:

$$\boldsymbol{H}^c = \sum_{k}^{K}\mathbf{R}^c_{:,:,k}, \tag{5}$$

where $\boldsymbol{H}^c \in \mathbb{R}^{W\times H}$ could be further normalized and resized to generate a heatmap that shows from which image regions features related to the particular output are extracted.

4. Attribution baseline

We first define two baseline properties based on the essence of Aumann–Shapley method itself. There are several reasonable definitions for feature missingness, such as making the output score drop to zero, drop significantly, or drop to a decision boundary. In our proposal, we use the first definition, *i.e.*, zero output, to represent feature missingness.

Property 1. Attribution baseline ought to express feature missingness of the particular output, *i.e.*, $f^c\Big(\overline{\mathbf{A}}^c\Big)\approx 0$.

According to the efficiency axiom (see Appendix A for detail information) of Aumann-Shapley values, it would be natural that the network output is zero given a baseline input which simulates the missingness of features related to the particular output. In such cases, the attribution computation given in Eq. (4) can be regarded as distributing the original particular output to the neuron features. This allows us to interpret the attributions as feature contributions of the particular output. The property can also be interpreted as the neutrality to the particular output. Only a baseline that can convey a complete absence of particular output features; then, when it is applied to compute neuron attributions, the integration (Eq. (4)) can "perfectly" sample from no feature to complete features. In addition, for most CNNs, a zero output can be produced by modifying the original feature maps, which also makes the baseline calculation easier to implement.

Property 2. Attribution baseline values ought to approach original feature values, *i.e.*, $\overline{A}^c_{i,j,k}\approx A_{i,j,k}$.

As an extension of Shapley values, Aumann–Shapley values are originally designed for infinite games where each value of the baseline approaches the original feature value. We can find that from Eq. (2) where $\Delta A_{i,j,k}$ implicitly affects the error term when calculating attribution scores. The basic idea of Aumann–Shapley values can be extended to CNNs to compute neuron attributions provided that the gap between the baseline and the original feature is minimized. For CNNs, we find that a big gap cause the attributions to be disturbed by neurons correlated with other outputs. This property can also be considered as the ability of decoupling output correlations, *i.e.*, the baseline should not affect the non-targeted outputs: $f^{c\prime}\Big(\overline{\mathbf{A}}^c\Big)\approx f^{c\prime}(\mathbf{A})$, $c^{'} \neq c$. Decoupling correlations between the particular output and the other outputs allows us to reduce the impact of the output correlations on attribution scores. Although an exact decoupling is difficult due to the prohibitively large search space which consists of tens of thousands of neurons in a convolutional layer, our attribution property implicitly provides the decoupling capability. Because the high similarity between the baseline and the original feature maps could minimize the baseline's influence on non-targeted outputs.

We can find that Property 1 can be satisfied by zero and noise baselines, but none of the baselines stated in Sec. 2.1 can satisfy Property 2. To generate baselines satisfying these inherent requirements of Aumann–Shapley attributions, we next propose two baseline selection methods, *i.e.*, optimization baseline and quadratic approximation baseline.

4.1. Optimization baseline

The baseline properties can guide the definition of selecting baselines–a baseline calculation method ought to retain all possible neurons that are not strictly correlated with the particular output and to remove the remaining neurons. Naturally, the baseline calculation method can be formulated as a subset selection problem:

$$\mathbb{S}^* = \arg\min_{\mathbb{S}} \left(f^c(\mathbf{A}(\mathbb{S})) \right)^2 + \lambda(|\mathbb{S}| - WHK)^2, \tag{6}$$

where $\mathbb{S}$ is a subset of the set of all neurons in the layer. The score $f^c(\mathbf{A}(\mathbb{S}))$ is the particular output if we only consider a part of neurons in the set $\mathbb{S}$. The operator $|\cdot|$ means the cardinality of a set. The parameter λ is used to balance the influence of similarity between the original feature maps and the baseline. $\mathbb{S}^*$ is used to indicate the neurons to be retained. For the objective function, the first term is designed for Property 1 and the second term is for Property 2. Although the subset selection is intuitive, solving it is NP-hard and clearly prohibitive for thousands of neurons in CNNs.

Inspired by research of adversarial samples [29] and Lasso regression [30], we transform the discrete problem into a continuous regularized energy minimization task. Neuron features of CNNs have separate and unique characteristics along the spatial and channel dimensions. In the spatial dimension, each neuron in one feature map corresponds to the image region whose size is the receptive field of the layer. Along the channel dimension, different neurons definitely represent different features, because each channel is determined by the operation of a 3D convolutional kernel. According to these two characteristics, we design the objective function from the two dimensions of space and channel separately, so as to reduce the difficulty of the optimization in Eq. (6). In particular, we formulate Eq. (6) as:

$$\boldsymbol{S}^{c*} = \arg\min_{\boldsymbol{s}} \left(f^c(\boldsymbol{S} \odot \mathbf{A}) \right)^2 + \lambda \|\boldsymbol{s} - \vec{1}\|_2^2, \tag{7}$$

$$\boldsymbol{z}^{c*} = \arg\min_{\boldsymbol{z}} \left(f^c(\boldsymbol{z} \odot \mathbf{A}) \right)^2 + \mu \|\boldsymbol{z} - \vec{1}\|_2^2, \tag{8}$$

where the spatial mask matrix $\boldsymbol{S} \in [0,1]^{W \times H}$ can be applied to the original feature maps to remove features spatially corresponding to the particular output. Note that, each element of the mask is a continuous weight. Similarly, the channel mask vector $\boldsymbol{z} \in [0,1]^K$ is applied to the channel dimension of the feature maps to remove channel features. The operator $\odot$ means Hadamard product. For the Hadamard product, the mask should be broadcast to the same size as the tensor for element-wise multiplication. The parameter λ and μ is used to balance the scale of two terms. For the simplicity of the following equations, we use the vector $\boldsymbol{s} = [S_{1,1}, \ldots, S_{W,H}]^T$ to denote the flattened $\boldsymbol{S}$. $\vec{1}$ is a vector of ones. The operator $\|\cdot\|_2$ means 2-norm of a vector.

With the resulting masks, the optimization baseline $\overline{\mathbf{A}}^{c*}$ can be generated by element-wisely multiplying original feature maps and the generated mask. In real implementation, we broadcast two masks to the size of feature maps and access element-wise minimum of two masks for the final multiplication with original feature maps.

4.2. Quadratic approximation baseline

Optimizing Eqs. (7) and (8) is computationally expensive because the gradient-based optimization is performed iteratively where we have to compute the gradients at each iteration to update the masks. The lower layer we select, the longer one iteration takes to compute the gradients according to the back-propagation process. To solve this problem, we approximate Eqs. (7) and (8) as quadratic optimization problems where we compute the gradients only once.

The approximation process of Eqs. (7, 8) are very similar. For the sake of simplicity, we only discuss one of them. Eq. (7) can be rewritten as a multivariate function, *i.e.* $f^c(\boldsymbol{S} \odot \mathbf{A}) = f^c(S_{1,1}\mathbf{A}_{1,1,:}, \ldots, S_{W,H}\mathbf{A}_{W,H,:})$. Then, we use the first order Taylor decomposition to decompose $f^c(\boldsymbol{S} \odot \mathbf{A})$ into a linear form: $f^c(\boldsymbol{S} \odot \mathbf{A}) \approx f^c(\mathbf{A}) + \sum_{i,j}(S_{i,j} - 1) g(\mathbf{A}_{i,j,:} \odot \partial f^c / \partial \mathbf{A}_{i,j,:}|_{\mathbf{A}})$, where g is a function that maps the Hadamard product between $\mathbf{A}_{i,j,:}$ and the gradient thereof to a scalar. By substituting this decomposition into Eq. (7), the objective function can be approximated with a quadratic optimization:

$$\boldsymbol{s}^{c*} = \arg\min_{\boldsymbol{s}} \boldsymbol{s}^T (\boldsymbol{q}\boldsymbol{q}^T + \lambda \boldsymbol{I}) \boldsymbol{s} - 2\lambda \boldsymbol{s} \cdot \vec{1} + 2(f^c(\mathbf{A}) - q) \boldsymbol{s} \cdot \boldsymbol{q}, \tag{9}$$

where $\boldsymbol{q} = [g(\mathbf{A}_{1,1,:} \odot \partial f^c / \partial \mathbf{A}_{1,1,:}|_{\mathbf{A}}), \ldots, g(\mathbf{A}_{W,H,:} \odot \partial f^c / \partial \mathbf{A}_{W,H,:}|_{\mathbf{A}})]^T$. The sum of $\boldsymbol{q}$ is q. $\boldsymbol{I}$ is an identity matrix of size WH. The vector $\boldsymbol{s} = [S_{1,1}, \ldots, S_{W,H}]^T$ is generated by flattening $\boldsymbol{S}$. The operator "$\cdot$" is a dot product between two vectors. For multivariate Taylor expansion, the function g can be a summation function. The mask vector $\boldsymbol{z}^{c*}$ in Eq. (8) can be calculated analogously.

The original iterative objective function is reduced to a quadratic optimization problem that can produce an optimal solution, *i.e.*, a unique baseline. There are many numerical optimization algorithms can be used to solve a quadratic optimization problem such as L-BFGS-B [31] with a box constrain range from 0 to 1. Solving Eq. (9) only needs to calculate the gradients between the particular output and feature maps once that is much faster than solving Eq. (7). With the baseline, we can calculate neuron attributions $\mathbf{R}^c$ and generate a heatmap in the current layer to understand the network behavior.

5. Experiments

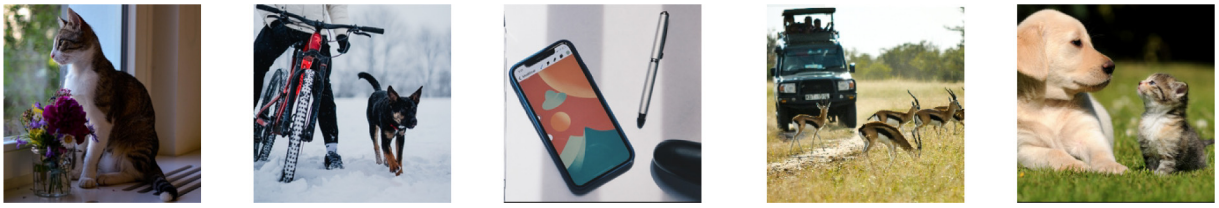
The first four experiments are validations of our proposed method: 1) evaluation of baseline properties (Sec. 5.1), 2) comparison to other baseline selections (Sec. 5.2), 3) comparison to other attribution methods (Sec. 5.3), and 4) pointing game evaluation (Sec. 5.4). Then, we implement two useful applications of neuron attributions: 5) attribution-based pruning method (Sec. 5.5) and 6) network repairing (Sec. 5.6).

The experiments are conducted using GoogLeNet [32] pre-trained on the ImageNet [33] dataset which has become an important backbone network in a variety of vision tasks. The iterative number T in Eq. (4) is set at 50, because we find that it is sufficient to make the attributions approximately add up to the particular output score in all tested cases, *i.e.*, roughly satisfying the "efficiency" axiom in Appendix A. One difficulty in generating a quadratic approximation baseline is to balance the different terms. Basically, these parameters (*i.e.*, λ and μ) can be selected in such a manner that, for reasonable mask size, the regularization term is about $1/10$ of the score term. In our experiments, we set the parameters $\lambda = 0.08 \min(q^2, f^c(\mathbf{A})^2)/(WH)$ for the spatial mask; $\mu = 0.1 \min(q^2, f^c(\mathbf{A})^2)/K$ for the channel mask. Our experiments are implemented on AMD Ryzen7 1700X 8-Core CPU and nVIDIA Titan RTX GPU. For baseline assessment and attribution evaluation, we randomly select 1000 images from the ImageNet validation dataset.

Fig. 1 shows images used for visually demonstrating experimental results that were downloaded from https://unsplash.com/ and https://github.com/distillpub/, licensed under Creative Commons Attributions CC-BY 4.0. Classification logits and class names generated by GoogLeNet are shown in Table 1.

5.1. Evaluation on baseline properties

According to the discussion in Sec. 4, the baseline of Shapley values should satisfy two properties. In this section, we evaluate the extent to which our baselines, *i.e.*, optimization baseline (OPT) and quadratic approximation baseline (QA), can satisfy the two baseline properties. We first calculate baselines of all tested layers (*i.e.*, from the mixed3a to mixed5b of GoogLeNet) using the randomly selected 1000 images from ImageNet validation dataset. The optimization baselines are generated using Adam optimizer [34]. We early stop updating when the loss variation less than 5% within 10 epochs. Quadratic approximation



(a) (b) (c) (d) (e)

Fig. 1. Images used for visual evaluation.

baselines are calculated using L-BFGS-B algorithm [31]. The average runtime of generating baselines and the standard deviations of runtime are shown in Table 2. The runtime depends on the depth of layer and the number of features. We can find that generating a QA baseline is much faster than generating an OPT baseline either on CPU or GPU.

For Property 1 (*i.e.*, the particular output approaches zero), we forward the calculated baselines into GoogLeNet to produce the scores of the particular outputs. The mean values of output scores of QA and OPT are 0.01 and 0.00, respectively. Both QA and OPT baselines can produce near-zero scores to represent the feature missingness indicating the satisfaction of Property 1.

For Property 2 (*i.e.*, the baseline approaches the original feature), we first calculate the mean values of spatial and channel masks, where 0 means complete masking and 1 means complete preservation of feature maps. As shown in Table 3, we find both of QA and OPT channel masks can keep most features, because both mean values are higher than 0.7. For spatial masks, the mean value of the OPT mask is 0.08 higher than that of QA, indicating that the OPT spatial mask is better. Overall, all these masks can keep more than 50% of features on average, which means that the baselines generated using these masks can effectively approach the original feature maps indicating the satisfaction of Property 2. To give an intuition of the relationship between resulting baselines and original feature maps, we also calculate Pearson correlation coefficients (PCCs) between original feature maps and baselines, as shown in Table 4. For the zero vector, the machine epsilon was added. The OPT baseline is closer to the original feature maps, but the correlation of QA also achieves 0.74. The correlation scores of QA and OPT numerically demonstrate that both baselines can reasonably approach the original feature maps.

5.2. Comparison to other baseline selections

In this section, we compare Aumann–Shapley (AS) attribution results generated using eight choices for neuron attribution baselines, *i.e.*, Uniform Noise [16], Gaussian Noise [16], Zero [9,13,14], MaxDistance [16], Expectation [6], Neutral [17], QA (ours), and OPT (ours) baselines. For Neutral baselines, we find that top-1 outputs of GoogLeNet are mostly distributed between 0.5 and 0.7. But the original setting of the neutrality value of their method is 0.5 that may be not suitable for the decision boundary of GoogLeNet. Therefore, to better satisfy the attributes of separating decision boundary defined in [17], we set the neutrality value at 30% of the mean score of one class, where output scores are calculated using the validation set. For other compared baselines, we follow the original ideas to reimplement them.

Without satisfying the two properties, the attribution explanations may be misleading, because attributions are affected by neurons not actually related to the particular output, especially in the higher layers (*e.g.*, mixed4 and mixed5), where feature maps have a significant spatial correlation with the target region. Fig. 2 shows the heatmaps generated using Aumann–Shapley attributions with different baselines. We can find that larger scores are assigned to some neurons corresponding to irrelevant regions using other baselines. These misleading explanations prevent us from truly understanding the network behaviors. For example, in the first row of Fig. 2, image regions outside the vase are also highlighted. Our baselines (*i.e.*, QA and OPT), which satisfy the two properties, can reduce the interference of irrelevant features, thereby generating attribution scores that more accurately express the feature contributions of the particular output.

On the other hand, although attribution heatmaps are consistent with human intuition, we still cannot conclude that these attribution scores are veritable and accurate. Because the object cognition from the perspective of a CNN is very different from human beings. Heatmaps which agree with the human intuition can only be used as an auxiliary tool to evaluate attribution results. Therefore, we further run these methods on two quantitative metrics, *i.e.*, *robustness* and *sensitivity-n* [35,36] which can be applied to evaluate neuron attributions. We do not select sanity check [37] and remove and retrain (ROAR) [38] metrics which are widely used for pixel attribution assessments, because they are difficult to be applied to evaluate neuron attributions.

Robustness metric, also called degradation test [36] or deletion region representativeness [15], is originally designed to evaluate pixel attributions by counting the change of the target score when pixel values are replaced with baseline values according to their contributions. This metric can be directly utilized to evaluate neuron attributions. Given the attribution scores, the neurons of the feature maps $\mathbf{A}$ are ranked. At each step, the highest ranked neurons in the set $\mathbb{S}$ are replaced. Then, the modified feature maps $\mathbf{A}_{\mathbb{S}}$ is forwarded into the network to predict a new classification score of the particular output, where $\mathbf{A}_{\mathbb{S}}$ means that the neurons in the set $\mathbb{S}$ are replaced by baseline values while others remain as they are. A larger increase of the changed particular output score $\left(f^c(\mathbf{A}) - f^c(\mathbf{A}_{\mathbb{S}})\right)/f^c(\mathbf{A})$ indicates better attribution results. Since the logit score may be negative, the score variation can be over 100%. We calculate attribution scores using the 1000 images selected in Sec. 5.1 on their labeled classes. Then, we replace top ranked $WH/2$ neurons at each step until 5% of neurons are replaced based on the attribution scores. Fig. 3 shows the score changes with all tested baselines in several layers of GoogLeNet. Our method outperforms other methods in all layers that means our method can quickly and accurately detect critical neurons. Generally, AS-OPT is slightly better than AS-QA; however, the optimization with gradient descent could be time-consuming. From the experimental results, we can also find that the baseline selection indeed have a huge impact on attribution results.

Table 1
Class information of images in Fig. 1.

Image	Fig. 1(a)	Fig. 1(b)	Fig. 1(c)	Fig. 1(d)	Fig. 1(e)
Top class/logit	Tabby/18.22	Kelpie/12.36	Telephone/14.73	Gazelle/20.51	Labrador retriever/11.32
Other class/logit	Vase/7.11	Bike/11.31	Ballpoint/8.51	Jeep/6.54	Tiger cat/4.45

Table 2
Runtime comparison between QA and OPT baselines.

	QA (CPU)	OPT (GPU)	OPT (CPU)
Mean runtime (s)	3.92	32.78	141.22
Std of runtime	8.72	18.71	107.11

Table 3
Mean values of generated spatial and channel masks where 0 means complete masking and 1 means complete preservation of feature maps.

	QA Spatial	OPT Spatial	QA Channel	OPT Channel
Mean mask values	0.52	0.6	0.70	0.72
Std of mask values	0.09	0.11	0.08	0.11

Table 4
PCCs between feature maps and baselines.

	QA	OPT
Mean PCC	0.74	0.87
Std of PCC	0.07	0.05

Although Aumann–Shapley method theoretically satisfies several desirable properties as discussed in Appendix A, a baseline without considering the essence of the approximation method could seriously degrade the attributions.

Sensitivity-n metric is designed to assess relationship between attributions and output scores systematically that can be considered as a general evaluation of the efficiency axiom (Appendix A). Similar to the robustness metric, although the original sensitivity-n is used to evaluate the pixel attributions, it can also be applied to evaluate the neuron attributions. Specifically, for any coalitions of neurons of size n, an attribution method satisfies sensitivity-n if the sum of these n neuron attributions is equal to the amount of the score variation caused by replacing these neurons with the baseline. The attribution scores are also calculated using the randomly selected 1000 images on their labeled classes. For a coalition $\mathbb{S}$ of size n, the modified feature maps $\mathbf{A}_{\mathbb{S}}$ is generated by replacing these neuron features with baseline values. Given a coalition $\mathbb{S}$ containing n randomly selected indices, sensitivity-n measures the Pearson correlation coefficient (PCC):

$$PCC\left(\sum_{\mathbb{S}}\mathbf{R}^c_{\mathbb{S}}, f^c(\mathbf{A}) - f^c(\mathbf{A}_{\mathbb{S}})\right), \tag{10}$$

where $\mathbf{R}^c$ is the neuron attributions. We implement the experiment by selecting the number of replaced neurons in *log-scale* between 1 and 95% of all neurons, because our method outperforms other methods given more than 10% neurons replaced in most layers. For each size n, we estimate the PCC by randomly sampling 200 coalitions of neurons in one convolutional layer. Fig. 4 shows the sensitivity-n scores with all tested attribution methods in several layers of GoogLeNet. To compare sensitivity-n results clearly, we further compute the average of log-scale sensitivity-n results, as shown in Table 5. These results show that our method outperforms all tested methods globally. As a systematic evaluation metric of the efficiency axiom, these better sensitivity-n results indicate that our method is able to systematically distribute attributions based on output score variations.

5.3. Comparison to other attribution methods

In this section, we report quantitative experiment evaluating our method alongside Saliency [20], GradxInput [21], SmoothGrad [22], EpsilonLRP [23], DeepLIFTRescale, DeepLIFTReveal [14,21], and DeepSHAP [28] using *robustness* and *sensitivity-n*. Although some methods were originally designed to calculate pixel attributions, they can be extended to calculate neuron attributions with slight changes.

Robustness metric results illustrate that our method can detect most contributing neuron features fastest, as shown in Fig. 5. We find the results in the mixed5b layer that is next to the output layer are nearly equivalent. Based on the discussion of the influence of network behavior linearization on attribution methods [35], this equivalence should be due to the weak nonlinearity of the sub-network from the last few layers to the output. The results shown in Fig. 3 and Fig. 5 indicate the decisive influence of baseline selections on attribution results. Some of AS

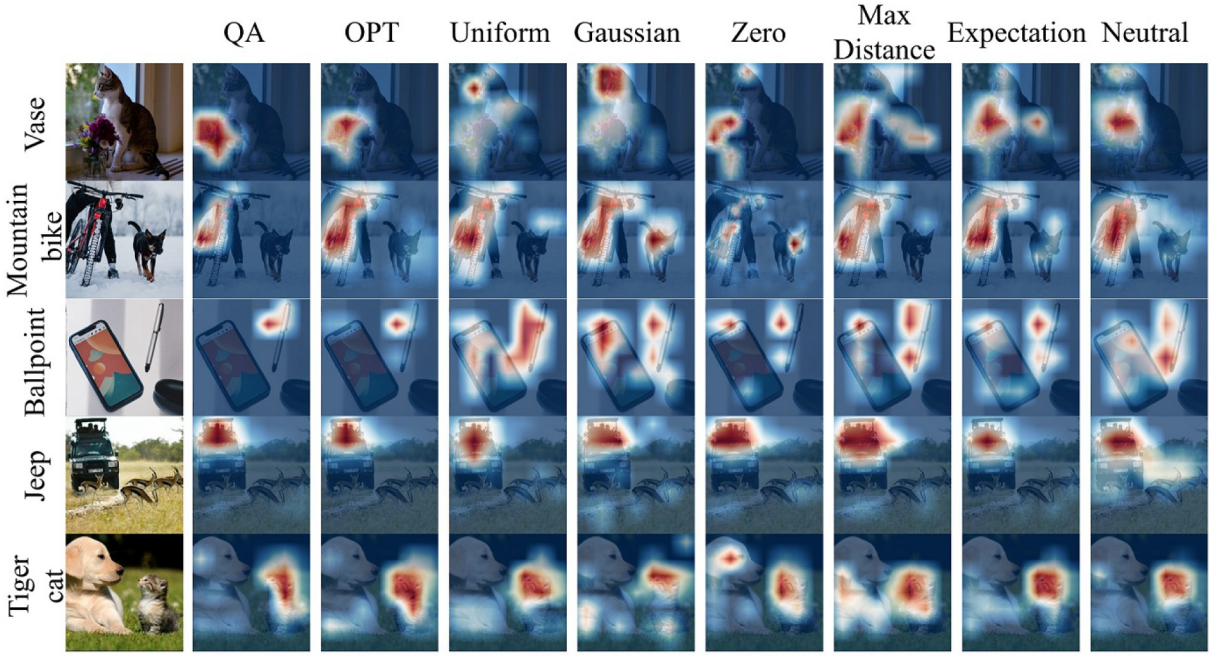


Fig. 2. Attribution heatmap comparison using Aumann–Shapley attributions with different baselines. Neuron attributions are calculated in the mixed4e layer of GoogLeNet. From left to right: original image, QA (ours), OPT (ours), Uniform Noise [16], Gaussian Noise [16], Zero [9,13,14], MaxDistance [16], Expectation [6], and Neutral [17]. The left most column indicates the targeted classes of these images.

attribution scores generated using other baseline selections result in less robustness than other attribution methods, whereas AS attribution calculation with our baseline outperforms all the other attribution methods.

Sensitivity-n metric results are shown in Fig. 6 using different attribution methods. Numerical results are shown in Table 6. These results show that our method outperforms all tested methods except GradxInput and LRP that perform better if the number of replaced neurons is small. Theoretically, gradient information could better express the local correlationship between numerical changes of neurons and output variations than the marginal contributions of Shapley values. But DeepLIFT and DeepSHAP that are also backpropagation-based do not produce good sensitivity-n scores. We speculate that their modifications of the backpropagation rule may deteriorate the nature of original gradients.

Although sensitivity-n can thoroughly evaluate whether the calculated neuron attributions systematically reflect neuron contributions to the output [35,36], we also find that this may be not a perfect systematic evaluation metric. The baseline heavily affects sensitivity-n results, *e.g.*, the sensitivity-n of our method is always better than others when replacing more than 20% neurons, because many neurons in our baseline approach the original ones. In addition, in the last layer mixed5b where the influence from the non-linear function is minimal, most methods produce attribution scores that almost satisfy sensitivity-n. However, we cannot assess the impact of the degree of non-linearity on the sensitivity-n results, and thus cannot confirm how much the good sensitivity-n results of different attribution methods are affected by the network structure.

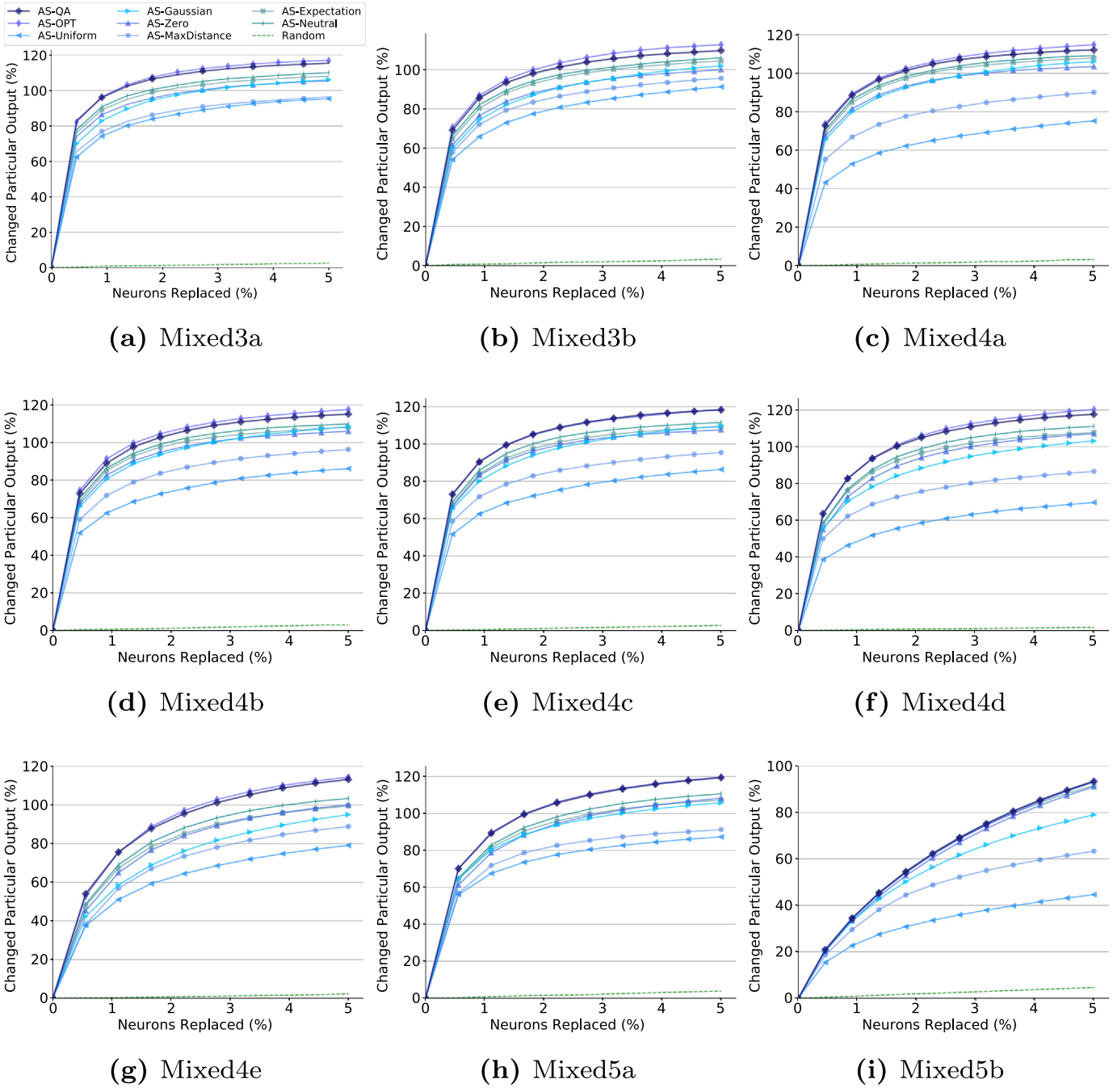


Fig. 3. The robustness results of the AS attributions with different baselines in different layers.

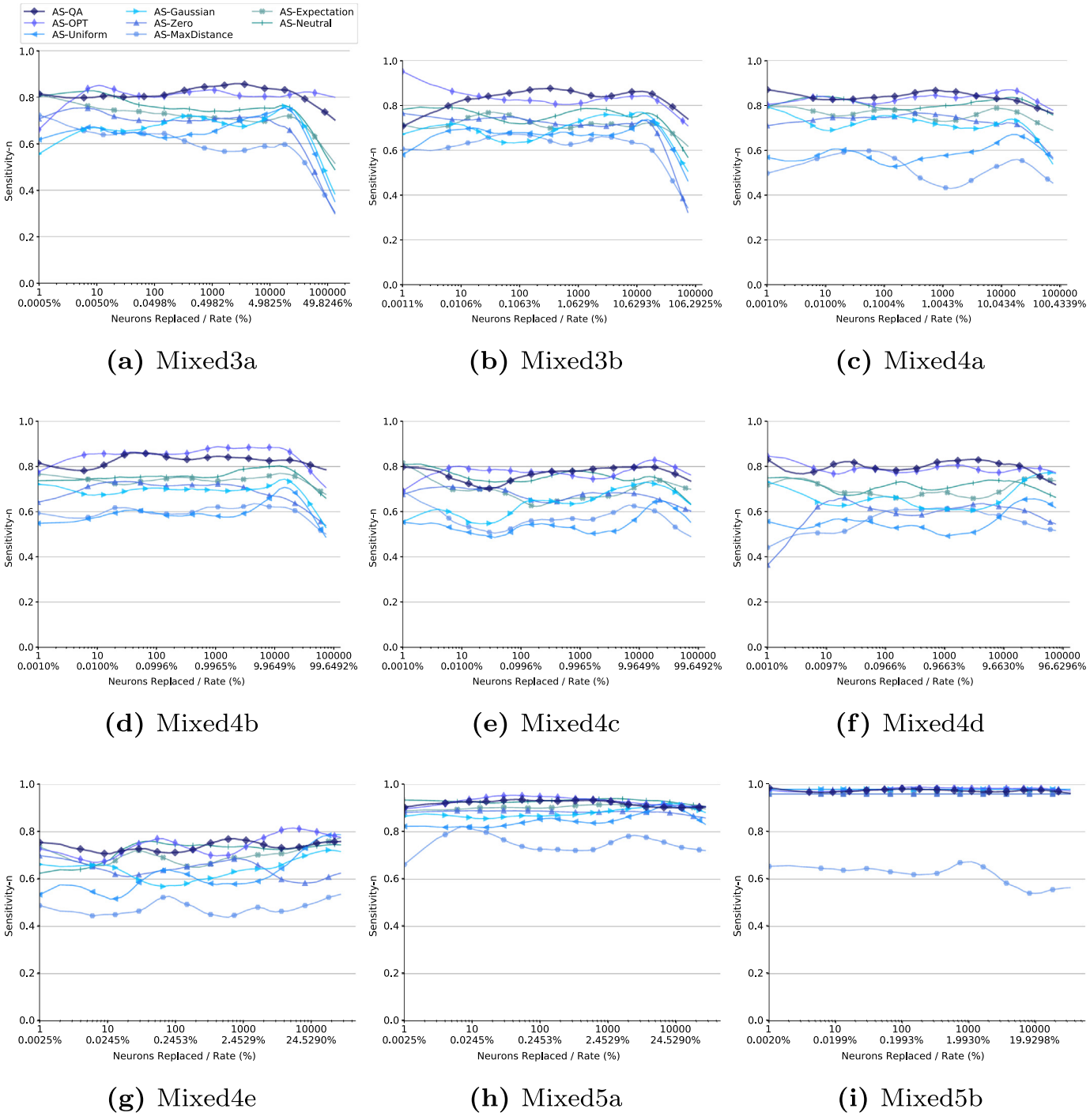


Fig. 4. The sensitivity-n results of the AS attributions with different baselines in different layers.

5.4. Pointing game evaluation

In this section, we implement the pointing game [39] on all tested methods that evaluate whether attribution scores can correlate with semantics in input images. Specifically, we generate attribution heatmaps in the mixed4e layer and count whether the maximum value in the heatmap can point out the object. The overall accuracy is the number of successful points over all results. Table 7 shows averaged results for this metric on PASCAL VOC [40] test set and COCO [41] validation set. As stated in [39], we also conduct the evaluation over the full data and a subset of difficult images. The experiment results show our method is competitive in this benchmark especially

Table 5
Sensitivity-n evaluation on different baselines in all layers.

	Layers									
Methods	3a	3b	4a	4b	4c	4d	4e	5a	5b	mean
AS-QA	0.80	0.82	0.83	**0.83**	**0.77**	0.78	**0.75**	0.91	0.97	**0.83**
AS-OPT	**0.82**	**0.84**	**0.85**	**0.83**	**0.77**	**0.80**	0.74	**0.93**	**0.99**	**0.84**
AS-Uniform	0.65	0.67	0.59	0.60	0.54	0.56	0.62	0.85	0.98	0.67
AS-Gaussian	0.67	0.69	0.71	0.69	0.64	0.66	0.64	0.88	0.98	0.73
AS-Zero	0.67	0.70	0.73	0.69	0.67	0.60	0.64	0.88	0.96	0.73
AS-MaxDistance	0.59	0.62	0.52	0.60	0.57	0.56	0.48	0.75	0.62	0.59
AS-Expectation	0.71	0.72	0.76	0.74	0.69	0.70	0.70	0.91	0.96	0.77
AS-Neutral	0.75	0.75	0.81	0.76	0.75	0.71	0.72	0.93	0.98	0.80

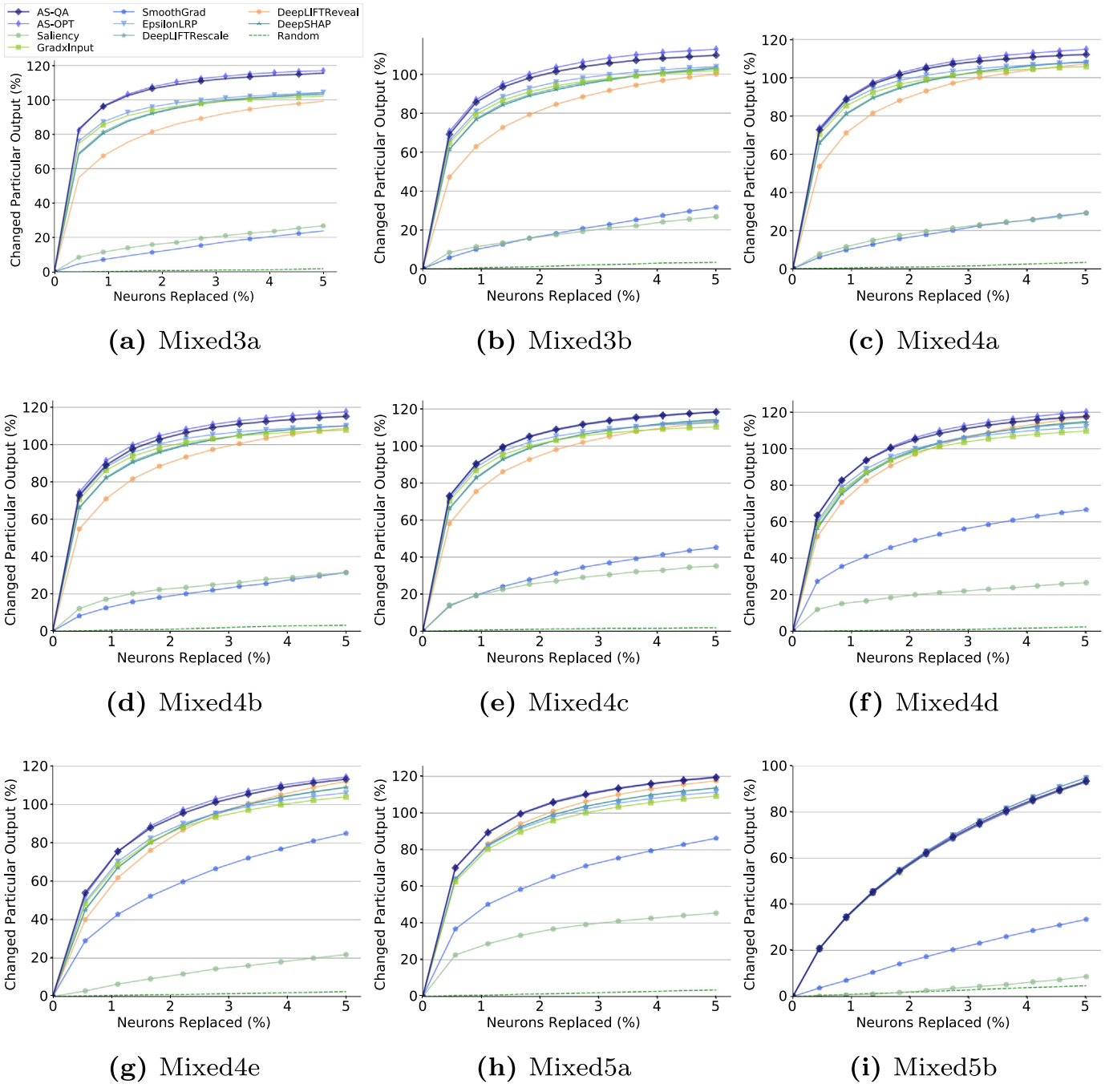


Fig. 5. The robustness results of the attribution methods in different layers.

on difficult images, because our proposal is always the fastest at identifying the most important neurons as discussed in the robustness experiments (Sec. 5.2 and 5.3).

5.5. Attribution-based pruning method

Designing a reasonable CNN structure may be difficult for non-machine learning experts; therefore, fine-tuning on an open-sourced and well-designed network is a common choice in practice, *e.g.*, fine-tuning YOLOv3 [42] to detect mangos in agronomic applications [43]. The pre-trained YOLO can detect 80 classes of objects where only apples and oranges are correlated with mangos. Although high accuracy can be achieved by retraining the original YOLOv3-tiny, it is more useful to find a low-power-consumption and small-size subnetwork that can detect one class of fruits accurately. Attribution scores can find the critical neurons, thereby detecting convolutional kernels related to the particular output class. Thus, we can prune unnecessary kernels based on attribution scores to get a small network for single class detection which is particularly useful in orchard applications.

We apply AS-QA to the detection network following the process discussed in [43] to find kernels correlated with the target class in each layer. According to the error increases of removing kernels, the pruning rates of 11 convolutional layers of YOLOv3-tiny are set to 0, 0, 0, 50, 50, 75, 87.5, 75, 75, 75, and 87.5%, which leads to 91.1% reduction of the entire network size. We freeze the weights in the first two layers and fine-tune other layers on the mango dataset. To compare with the experimental results of the previous study [43], we also use F1-score to evaluate the results. The thresholds of Intersection over Union (IoU), object score, and classification

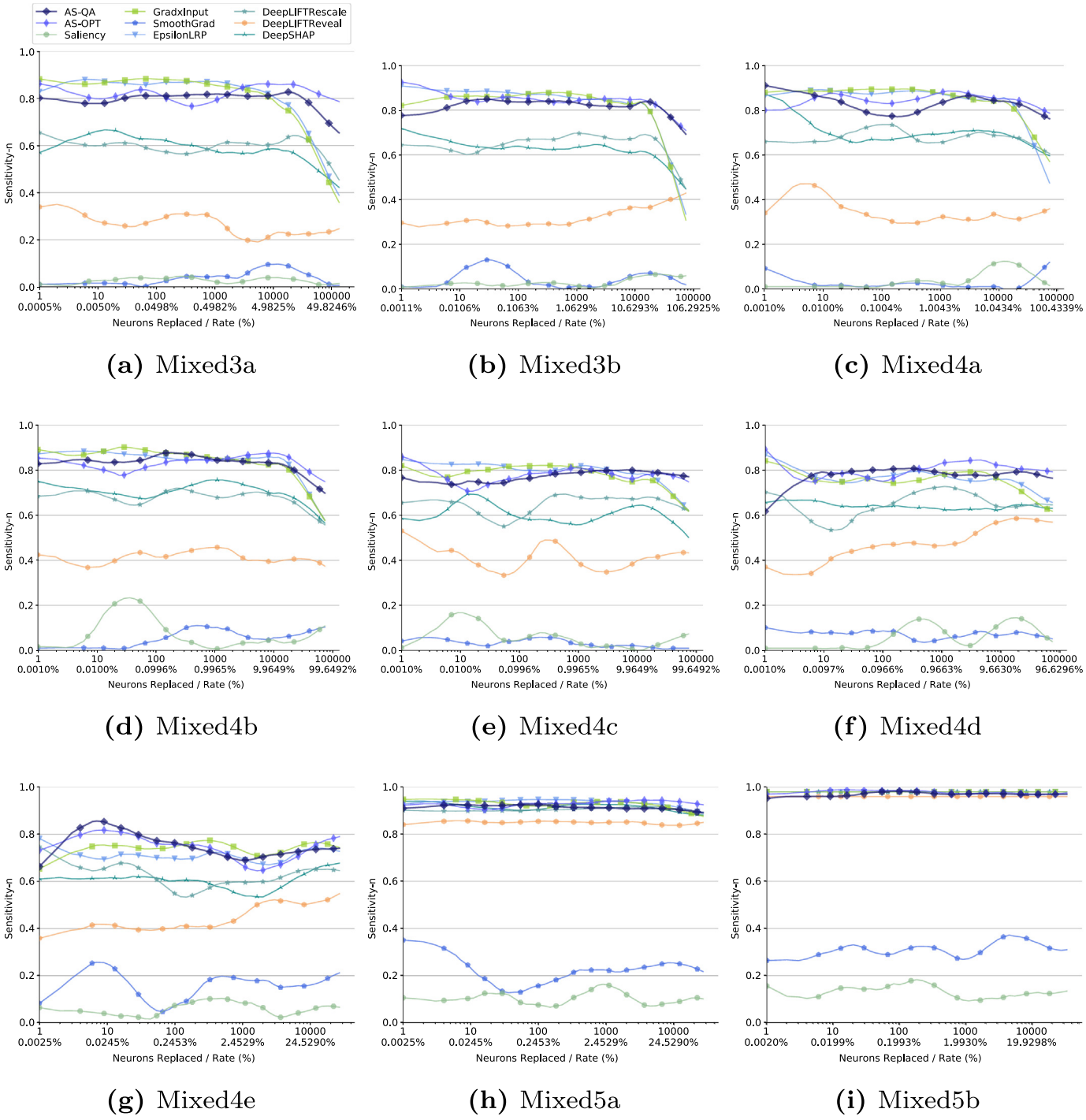


Fig. 6. The sensitivity-n results of the attribution methods in different layers.

score are all set to 0.5. Results of F1-score, trained weight size, and GFLOPs are shown in Table 8. Our pruned network, *i.e.*, YOLOv3-tiny (pruned with AS-QA), reduces the GFLOPs from 8.3 by 72.3% to 2.3 and further improves the accuracy achieved in [43]. Using AS-QA attributions allows us to remove irrelevant convolutional kernels more effectively, so as to better choose the pruning rates and kernels. This technique can prune a well-designed and large-scale network into a small network to detect single class of objects that can be deployed on mobile devices, *e.g.*, embedded micro-controller units. Moreover, it is straightforward and easy to use even for non-machine learning engineers.

5.6. Detecting vulnerable neurons

An adversary method can mislead the true output of CNNs into arbitrary result by just adding imperceptible perturbations on the original images. Recently, Ghorbani et al. [26] proposed to use channel-level attributions to detect channels mostly influenced by adversary samples. In this experiment, we further extend their idea to neuron level and find neuron-level defense of adversary samples can be more efficient than that of channel level.

We apply neuron attributions to identify the neuron features most related to the adversarial attacks. We first select a general iterative

Table 6
Sensitivity-n evaluation on different attribution methods in all layers.

	Layers									
Methods	3a	3b	4a	4b	4c	4d	4e	5a	5b	mean
AS-QA	0.80	0.82	0.83	0.83	0.77	0.78	**0.75**	0.91	0.97	**0.83**
AS-OPT	**0.82**	**0.84**	**0.85**	0.83	0.77	**0.80**	0.74	0.93	**0.99**	**0.84**
Saliency	0.03	0.03	0.04	0.08	0.06	0.06	0.06	0.10	0.13	0.07
GradxInput	0.80	0.81	**0.85**	**0.84**	0.78	0.75	0.74	0.93	0.98	**0.83**
SmoothGrad	0.04	0.05	0.02	0.06	0.03	0.07	0.16	0.22	0.31	0.11
EpsilonLRP	0.81	0.82	0.84	**0.84**	**0.79**	0.76	0.71	**0.94**	0.98	**0.83**
DeepLIFTRescale	0.59	0.65	0.68	0.68	0.65	0.65	0.62	0.90	0.98	0.71
DeepLIFTReveal	0.28	0.34	0.36	0.43	0.43	0.50	0.46	0.87	0.98	0.52
DeepSHAP	0.59	0.62	0.69	0.70	0.61	0.64	0.60	0.91	0.98	0.70

Table 7
Pointing game evaluation. Averaged accuracy on the pointing game benchmark over all data and a subset of difficult images.

Method	VOC	COCO
AS-QA	87.1/77.9	58.1/52.6
AS-OPT	87.7/**78.2**	58.4/**53.7**
AS-Uniform	80.4/66.7	52.2/47.1
AS-Gaussian	83.9/68.9	54.1/48.4
AS-Zero	85.6/72.8	55.9/50.1
AS-MaxDistance	81.1/68.9	54.5/47.4
AS-Expectation	85.9/76.0	55.4/52.1
AS-Neutral	87.1/76.1	57.5/52.2
Saliency	72.3/49.4	42.3/37.6
GradxInput	86.8/75.7	57.7/51.1
SmoothGrad	78.2/59.1	49.7/43.4
EpsilonLRP	**88.9**/76.5	**58.8**/51.7
DeepLIFTRescale	86.1/76.2	57.2/51.6
DeepLIFTReveal	83.3/69.1	55.6/49.1
DeepSHAP	86.6/75.1	57.8/50.1

Table 8
Network performance (F1-score, weight size, and GFLOPs) for mango detection on test set images. Pruning the network based on AS-QA attributions further improves the network performance achieved in [43].

Networks	F1 score (%)	Weight Size (Mb)	GFLOPs
YOLOv3-tiny (pruned with AS-QA)	95.1	3.1	2.3
Pruned YOLOv3-tiny [43]	94.4	5.3	2.6
YOLOv3-tiny	94.0	34.8	8.3
YOLOv3	95.1	240.5	99.2
Faster R-CNN(VGG)	94.5	533.9	–
Faster R-CNN(ZF)	93.9	230.1	–

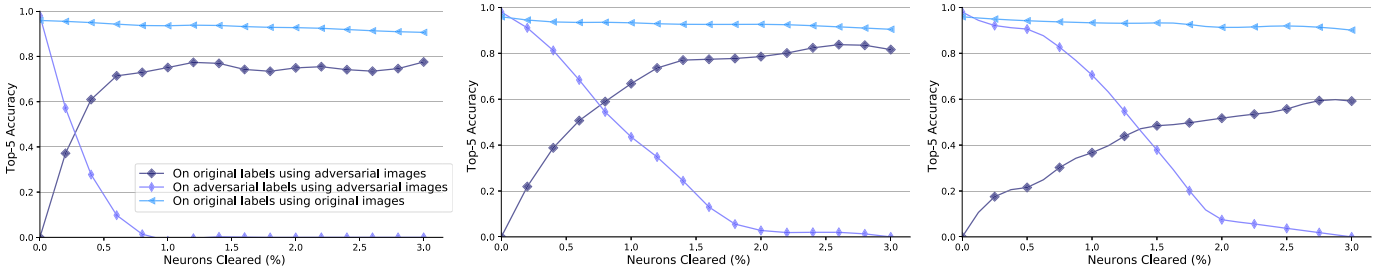
adversarial method [29] to generate attacks. Simply, we target the adversarial classes and generate perturbations using gradients iteratively on original images to mislead the network. We randomly select 20 classes of images from the first 700 classes in the ImageNet dataset, and then perturb these images by targeting 10 adversarial classes which are randomly selected from the last 300 classes.

High attribution scores mean that these neurons are more leveraged by the adversarial attack; thus, if we remove them, the accuracy on adversarial classes should drop heavily, whereas the accuracy on original labels is preserved. For evaluation, three metrics are used, *i.e.*, top-5 accuracies on original labels using adversarial images, on adversarial labels using adversarial images, and on original labels using original images. Then, as shown in Fig. 7, after zeroing out about 1.5% of neurons in these layers, the adversarial samples cannot mislead the network. When zeroing out 3% of neurons, the accuracy on the adversarial classes drops to near 0, *i.e.*, the adversarial information hidden in the perturbations cannot be extracted by the network. We also implement the experiment stated in [26] using channel attributions to repair the network, as shown in Fig. 8. All three results show that the defense of adversarial attack in the channel level is less effective than in the neuron level. For example, in Fig. 8(a), the attack disappear after 10% channels are removed, but the attack disappears after removing less than 1% of neurons in Fig. 7(a).

Generally, adversarial attacks are almost imperceptible to humans, that is, the attack may only affect a small number of neurons. Therefore, if these affected neurons can be detected and removed, we can eliminate the attack and reconstruct the original features effectively. This experiment suggests that neuron attributions can potentially offer a fast mechanism to repair fooled networks.

6. Conclusion

In this work, we discussed the significant influence of a baseline on attribution results and used Aumann–Shapley values as an axiomatic attribution method to compute neuron attributions. Based on the essence of Aumann–Shapley method itself, we proposed two baseline properties and designed a general objective function of generating baselines. Then, we approximated it as a quadratic optimization problem to generate the attribution baseline more efficiently. We demonstrated our method can be more accurate than other existing baseline selections and attribution methods using attribution heatmaps and three quantitative metrics, *i.e.*, robustness, sensitivity-n, and pointing game. Moreover, to show the practicability of neuron attributions, we applied attributions to network pruning and adversarial analysis. As research on this area continues, we would like to propose a metric to evaluate the error caused by the approximations and a better benchmark to assess attribution results systematically.



(a) Mixed4a (b) Mixed4d (c) Mixed5a

Fig. 7. Accuracy changes after zeroing out neurons most contributing to the adversarial classes.

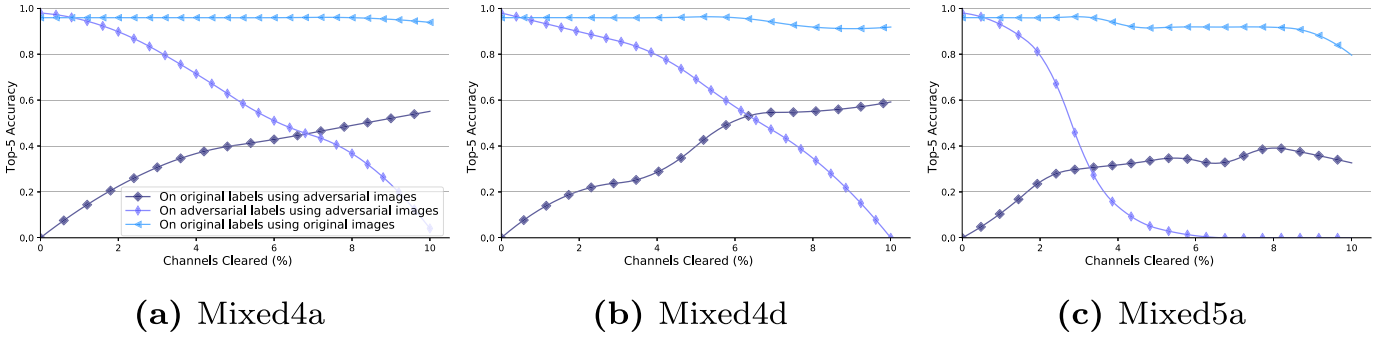


Fig. 8. Accuracy changes after zeroing out channels most contributing to the adversarial classes.

CRediT authorship contribution statement

Rui Shi: Conceptualization, Investigation, Methodology, Software, Visualization, Writing – original draft. **Tianxing Li:** Investigation, Methodology, Software, Writing – review & editing. **Yasushi Yamaguchi:** Conceptualization, Formal analysis, Investigation, Funding acquisition, Supervision, Writing – review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This research was partly supported by Japan Society for the Promotion of Science (JSPS KAKENHI) [grant number 20H04203].

Appendix A. Axioms of attribution methods

To better understand the merits of attribution methods, it is necessary to assess them with respect to several fundamental axiomatic properties. The main ideas of these axioms can already be found in [11] in more general and abstract form, see also [8,9], but we want to rephrase them in a way that optimally prepares readers to the reason of selecting Aumann–Shapley values. We compare attribution methods using the following five desirable axioms established in previous studies: 1) *efficiency*, 2) *implementation invariance*, 3) *null player*, 4) *symmetry*, and 5) *linearity*.

Axiom 1. Efficiency. An attribution method satisfies efficiency when the attribution scores add up to the particular output f^c given the feature maps $\mathbf{A}$, *i.e.*, $\sum \mathbf{R}^c = f^c(\mathbf{A})$. This axiom, also called conservation [7] or completeness [9], is regarded as a sanity check that attribution scores are comprehensive in a numeric sense.

Axiom 2. Implementation invariance. If the outputs of two networks are equal for all inputs, these two networks are functionally equivalent, despite in possibly very different implementations [9]. An attribution method that satisfies implementation invariance should produce identical contribution scores when applied to the functionally equivalent networks given the same input. The attribution explanation failing to satisfy this axiom is potentially sensitive to unessential aspects of the network.

Axiom 3. Null player. If some features are independent of the implementation of a network, then these feature attributions should be zero. If a feature does not contribute to the result is assigned with a non-zero value, the zero attribution value becomes meaningless for all non-contributing features.

Axiom 4. Symmetry. If the function implemented by a network depends on two features equally, *i.e.*, not on their order, then the attributions assigned to these two features should be equal. It seems natural to satisfy symmetry because if two features play the exact same importance, they should receive the same attribution.

Axiom 5. Linearity. Suppose that an integrated network f is a linear combination of two different sub-networks, *i.e.*, $f = af_1 + bf_2$, then the attributions for the network f should be the weighted sum of the sub-attributions for f_1 and f_2 with weights a and b. Specifically, the attribution method should preserve the linearity among interpreted networks. If the attribution method cannot produce linear-correlated results given a linear combination, it is impossible to assess whether the attributions are reasonable in the case of deep network combinations.

Aumann–Shapley method and most Shapley-based attribution methods can satisfy these axioms. This is a direct conclusion from past research [8]. These axioms that any attribution method should satisfy motivates our utilization of Aumann–Shapley method to attribution calculation.

References

[1] F. Hohman, H. Park, C. Robinson, D.H. Polo Chau, Summit: scaling deep learning interpretability by visualizing activation and attribution summarizations, IEEE Trans. Vis. Comput. Graph. 26 (2020) 1096–1106.
[2] R. Guerrero-Gómez-Olmedo, J.L. Salmeron, C. Kuchkovsky, LRP-based path relevances for global explanation of deep architectures, Neurocomputing 381 (2020) 252–260.
[3] A. Katzmann, O. Taubmann, S. Ahmad, A. Mühlberg, M. Sühling, H.-M. Groß, Explaining clinical decision support systems in medical imaging using cycle-consistent activation maximization, Neurocomputing 458 (2021) 141–156.
[4] G. Ras, M. van Gerven, P. Haselager, Explanation methods in deep learning: Users, values, concerns and challenges, Explain. Intepret. Model. Comput. Vis. Mach. Learn. 2018, pp. 19–36.
[5] M. Ancona, C. Oztireli, M. Gross, Explaining deep neural networks with a polynomial time algorithm for Shapley value approximation, Proceedings of the International Conference on Machine Learning (ICML) 2019, pp. 272–281 , URL: http://proceedings.mlr.press/v97/ancona19a.html.
[6] S.M. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions, Advances in Neural Information Processing Systems (NIPS) 2017, pp. 4768–4777.
[7] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, K.-R. Müller, Explaining nonlinear classification decisions with deep Taylor decomposition, Pattern Recogn. 65 (2017) 211–222.
[8] Y. Sun, M. Sundararajan, Axiomatic attribution for multilinear functions, Proceedings of the ACM Conference on Electronic Commerce 2011, pp. 177–178, https://doi.org/10.1145/1993574.1993601.
[9] M. Sundararajan, A. Taly, Q. Yan, Axiomatic attribution for deep networks, Proceedings of the International Conference on Machine Learning (ICML) 2017, pp. 3319–3328, https://doi.org/10.5555/3305890.3306024.
[10] L.S. Shapley, A value for n-person games, Princeton University Press, 2016.
[11] R.J. Aumann, L.S. Shapley, Values of Non-atomic Games, Princeton University Press, 1974 URL: http://www.jstor.org/stable/j.ctt13x149m.
[12] I. Kumar, S. Venkatasubramanian, C. Scheidegger, S. Friedler, Problems with Shapley-value-based explanations as feature importance measures, Proceedings of the International Conference on Machine Learning (ICML) 2020, pp. 5447–5456.
[13] M.D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, European Conference on Computer Vision, Springer 2014, pp. 818–833.

[14] A. Shrikumar, P. Greenside, A. Kundaje, Learning important features through propagating activation differences, in, Proceedings of the International Conference on Machine Learning (ICML) 2017, pp. 3145–3153.
[15] R.C. Fong, A. Vedaldi, Interpretable explanations of black boxes by meaningful perturbation, in, Proceedings of the IEEE International Conference on Computer Vision (ICCV) 2017, pp. 3429–3437.
[16] P. Sturmfels, S. Lundberg, S.-I. Lee, Visualizing the impact of feature attribution baselines, Distill 5 (2020), e22https://doi.org/10.23915/distill.00022.
[17] C. Izzo, A. Lipani, R. Okhrati, F. Medda, A Baseline for Shapley Values in mlps: From Missingness to Neutrality, 2021 arXiv:2006.04896.
[18] J. Haug, S. Zürn, P. El-Jiz, G. Kasneci, On baselines for local feature attributions, 2021 arXiv:2101.00905.
[19] M. Sundararajan, A. Najmi, The many Shapley values for model explanation, Proceedings of the 37th International Conference on Machine Learning, volume 119 of Proceedings of Machine Learning Research, PMLR 2020, pp. 9269–9278 , URL: https://proceedings.mlr.press/v119/sundararajan20b.html.
[20] K. Simonyan, A. Vedaldi, A. Zisserman, Deep inside convolutional networks: Visualising image classification models and saliency maps, Proceedings of the International Conference on Learning Representations (ICLR), 2014 , URL: http://arxiv.org/abs/1312.6034.
[21] A. Shrikumar, P. Greenside, A. Shcherbina, A. Kundaje, Not Just a Black Box: Learning Important Features through Propagating Activation Differences, 2017 arXiv:1605.01713.
[22] D. Smilkov, N. Thorat, B. Kim, F. Viégas, M. Wattenberg, Smoothgrad: Removing Noise by Adding Noise, 2017 arXiv:1706.03825.
[23] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, W. Samek, On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation, PLoS One 10 (2015).
[24] E. Strumbelj, I. Kononenko, An efficient explanation of individual classifications using game theory, J. Mach. Learn. Res. 11 (2010) 1–18.
[25] J. Castro, D. Gómez, J. Tejada, Polynomial calculation of the Shapley value based on sampling, Comput. Oper. Res. 36 (2009) 1726–1730.
[26] A. Ghorbani, J.Y. Zou, Neuron Shapley: Discovering the responsible neurons, Advances in Neural Information Processing Systems (NIPS) 2020, pp. 5922–5932 , URL: https://proceedings.neurips.cc/paper/2020/file/41c542dfe6e4fc3deb251d64cf6ed2e4-Paper.pdf.
[27] K. Dhamdhere, M. Sundararajan, Q. Yan, How important is a neuron, in, Proceedings of the International Conference on Learning Representations (ICLR), 2019 , URL: https://openreview.net/forum?id=SylKoo0cKm.
[28] H. Chen, S. Lundberg, S.-I. Lee, Explaining models by propagating shapley values of local components, Explainable AI in Healthcare and Medicine, Springer 2021, pp. 261–270.
[29] A. Kurakin, I. Goodfellow, S. Bengio, Adversarial Machine Learning at Scale, arXiv:1611.01236 2017.
[30] R. Tibshirani, Regression shrinkage and selection via the Lasso, J. R. Stat. Soc. 58 (1996) 267–288.
[31] C. Zhu, R.H. Byrd, P. Lu, J. Nocedal, L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization, ACM Trans. Math. Softw. 23 (1997) 550–560.
[32] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015https://doi.org/10.1109/cvpr.2015.7298594.
[33] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2009, pp. 248–255, https://doi.org/10.1109/CVPR.2009.5206848.
[34] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, Proceedings of the International Conference on Learning Representations (ICLR), 2015 , URL: http://arxiv.org/abs/1412.6980.
[35] M. Ancona, E. Ceolini, C. Öztireli, M. Gross, Towards better understanding of gradient-based attribution methods for deep neural networks, Proceedings of the International Conference on Learning Representations (ICLR), 2018 , URL: https://openreview.net/forum?id=Sy21R9JAW.
[36] K. Schulz, L. Sixt, F. Tombari, T. Landgraf, Restricting the flow: Information bottlenecks for attribution, Proceedings of the International Conference on Learning Representations (ICLR), 2020 , URL: https://openreview.net/forum?id=S1xWh1rYwB.
[37] R. Tomsett, D. Harborne, S. Chakraborty, P. Gurram, A. Preece, Sanity checks for saliency metrics, Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, 2020, pp. 6021–6029 , URL: https://ojs.aaai.org/index.php/AAAI/article/view/6064 https://doi.org/10.1609/aaai.v34i04.6064.
[38] S. Hooker, D. Erhan, P.-J. Kindermans, B. Kim, A benchmark for interpretability methods in deep neural networks, Advances in Neural Information Processing Systems, vol. 32, , 2019 , URL: https://proceedings.neurips.cc/paper/2019/file/fe4b8556000d0f0cae99daa5c5c5a410-Paper.pdf.
[39] J. Zhang, S.A. Bargal, Z. Lin, J. Brandt, X. Shen, S. Sclaroff, Top-down neural attention by excitation backprop, Int. J. Comput. Vis. 126 (2018) 1084–1102.
[40] M. Everingham, S. Eslami, L. Van Gool, C.K. Williams, J. Winn, A. Zisserman, The pascal visual object classes challenge: a retrospective, Int. J. Comput. Vis. 111 (2015) 98–136.
[41] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C.L. Zitnick, Microsoft COCO: Common Objects in Context, European Conference on Computer Vision, Springer 2014, pp. 740–755.
[42] J. Redmon, A. Farhadi, Yolov3: An Incremental Improvement, arXiv:1804.02767 2018.
[43] R. Shi, T. Li, Y. Yamaguchi, An attribution-based pruning method for real-time mango detection with yolo network, Comput. Electron. Agric. 169 (2020), 105214.