BEE: Metric-Adapted Explanations via Baseline Exploration-Exploitation

Oren Barkan[1*], **Yehonatan Elisha**[2*], **Jonathan Weill**[2], **Noam Koenigstein**[2]

[1]The Open University, Israel
[2]Tel Aviv University, Israel

Abstract

Two prominent challenges in explainability research involve 1) the nuanced evaluation of explanations and 2) the modeling of missing information through baseline representations. The existing literature introduces diverse evaluation metrics, each scrutinizing the quality of explanations through distinct lenses. Additionally, various baseline representations have been proposed, each modeling the notion of missingness differently. Yet, a consensus on the ultimate evaluation metric and baseline representation remains elusive. This work acknowledges the diversity in explanation metrics and baselines, demonstrating that different metrics exhibit preferences for distinct explanation maps resulting from the utilization of different baseline representations and distributions. To address the diversity in metrics and accommodate the variety of baseline representations in a unified manner, we propose Baseline Exploration-Exploitation (BEE) - a path-integration method that introduces randomness to the integration process by modeling the baseline as a learned random tensor. This tensor follows a learned mixture of baseline distributions optimized through a contextual exploration-exploitation procedure to enhance performance on the specific metric of interest. By resampling the baseline from the learned distribution, BEE generates a comprehensive set of explanation maps, facilitating the selection of the best-performing explanation map in this broad set for the given metric. Extensive evaluations across various model architectures showcase the superior performance of BEE in comparison to state-of-the-art explanation methods on a variety of objective evaluation metrics.

Code — https://github.com/yonisGit/BEE

1 Introduction

Deep learning models have demonstrated remarkable success across a spectrum of tasks in computer vision (He et al. 2016; Dosovitskiy et al. 2020; Carion et al. 2020), natural language processing (Vaswani et al. 2017; Devlin et al. 2018; Barkan et al. 2020d,c, 2021a; Touvron et al. 2023), recommender systems (He et al. 2017; Barkan and Koenigstein 2016; Barkan et al. 2020a, 2021b,c; Katz et al. 2022), and audio processing (Barkan et al. 2019, 2023c; Engel et al. 2020; Kong et al. 2020). Despite their accomplishments, these models frequently function as opaque systems, introducing challenges in comprehending their predictions. Consequently, the field of Explainable AI (XAI) has emerged, dedicated to developing methods that illuminate the decision rationale of machine learning models across diverse application domains (Simonyan, Vedaldi, and Zisserman 2013; Malkiel et al. 2022; Gaiger et al. 2023; Barkan et al. 2020b, 2023a, 2024a,b). In the context of computer vision, XAI techniques aim to generate explanation maps highlighting input regions responsible for the model's predictions (Selvaraju et al. 2017; Chefer, Gur, and Wolf 2021b). For example, Integrated Gradients (IG) (Sundararajan, Taly, and Yan 2017), which produces explanation maps by integrating gradients along a linear path between the input image and a baseline representation (acting as a reference representing missing information). Nevertheless, a challenge persists in selecting the appropriate baseline, as different types of baselines model missingness differently, resulting in variations in the explanation maps. Despite the exploration of various baselines in the literature, no consensus has emerged on the ultimate baseline representation (Ancona et al. 2017; Fong and Vedaldi 2017; Sturmfels, Lundberg, and Lee 2020). Another prominent challenge in XAI revolves around evaluating the effectiveness of the generated explanations. Various evaluation metrics have been proposed in the literature (Chefer, Gur, and Wolf 2021b; Petsiuk, Das, and Saenko 2018; Kapishnikov et al. 2019; Chattopadhay et al. 2018), each assessing the quality of explanation maps from different perspectives. As a result, distinct explanation metrics may promote different explanation maps, and currently, there is no universally agreed-upon evaluation metric for assessing the goodness of explanations. Acknowledging the diverse landscape of evaluation metrics and baseline representations, this paper introduces Baseline Exploration-Exploitation (BEE) - a path-integration method utilizing an exploration-exploitation (EE) mechanism to adapt the baseline distribution (and hence the resulting explanation) w.r.t. the specific metric of interest. This approach offers an effective way to address the diversity in metrics and baselines in a cohesive manner. BEE integrates on the intermediate representations (and their gradients) produced by different network layers, thereby generating explanation maps at multiple levels of abstractions and various scales. The

[*]These authors contributed equally.

key innovation of BEE lies in introducing randomness to the integration process by modeling the baseline representation as a random tensor. This tensor adheres to a mixture of baseline distributions learned through an offline pretraining phase. The pretraining phase, employs contextual EE of the baseline distribution to optimize performance on the specific metric at hand. Given a test instance (context), BEE generates a pool of candidate explanation maps produced from a corresponding set of baselines sampled from the pretrained baseline distribution in parallel. Subsequently, the explanation map that performs the best on the metric is selected. For further enhancement, BEE can continually employ contextual EE on the specific instance during inference, facilitating instance-specific finetuning of the pretrained baseline distribution w.r.t. the metric of interest. The effectiveness of BEE is demonstrated through extensive evaluation on various model architectures. The results showcase that the pretrained BEE consistently outperforms latest state-of-the-art explanation methods across all objective evaluation metrics. Furthermore, when inference-time finetuning is permitted, the finetuned BEE yields additional performance gains over the pretrained BEE, affirming its viability as a superior option.

Our contributions are summarized as follows: 1) We highlight and demonstrate the challenges arising from the absence of a universally agreed-upon evaluation metrics and baseline representations in XAI: Different metrics promote different explanation maps resulting from different types of baselines. 2) To address these challenges, we introduce BEE - a novel contextual EE-based path-integration method. BEE models the baseline representation as a random tensor sampled from a mixture of distributions, accommodating various types of baselines, and facilitating adaptive explanations w.r.t. the metric at hand. 3) Through extensive evaluation against 13 explanation methods, both the pretrained and finetuned BEE versions emerge as a new state-of-the-art in XAI, outperforming latest state-of-the-art methods across 8 evaluation metrics and 5 CNN and ViT architectures.

2 Related Work

Literature on explanation methods for CNNs has grown with several broad categories of approaches: perturbation-based methods (Fong and Vedaldi 2017; Barkan et al. 2023a), gradient-free methods (Zhou et al. 2018, 2016; Zeiler and Fergus 2014; Wang et al. 2020), and gradient-based methods (Selvaraju et al. 2017; Srinivas and Fleuret 2019), which include path-integration methods (Sundararajan, Taly, and Yan 2017; Xu, Venugopalan, and Sundararajan 2020; Barkan et al. 2023b; Elisha, Barkan, and Koenigstein 2024). The most relevant line of work to this paper are path-integration methods (Sundararajan, Taly, and Yan 2017; Xu, Venugopalan, and Sundararajan 2020; Barkan et al. 2023b). IG (Sundararajan, Taly, and Yan 2017) integrates over the interpolated image gradients. Blur IG (Xu, Venugopalan, and Sundararajan 2020) integrates gradients over a path that progressively removes Gaussian blur from the attributed image. Our method, provides two significant differences w.r.t. the aforementioned works: First, BEE extends the integrand beyond the gradient itself, incorporating information from both the internal network representations and their gradients. This capability facilitates the generation of explanation maps at multiple levels of abstractions and resolution and has proven effective (Barkan et al. 2023b). Second, through the introduction of an EE procedure, BEE can learn baseline distributions customized for the specific metric of interest. This characteristic enables BEE to draw a set of explanation maps resulting from the sampled baselines and select the best-performing one w.r.t. the given metric. Early explanation methods for transformer explainability involved leveraging the inherent attention scores to gain insights into the input (Carion et al. 2020). However, a challenge arises in combining scores from different layers. Simple averaging of attention scores for each token, for instance, tends to blur the signal (Abnar and Zuidema 2020). Chefer et al.(Chefer, Gur, and Wolf 2021b) introduced Transformer Attribution (T-Attr), a class-specific Deep Taylor Decomposition method in which relevance propagation is applied for positive and negative attributions. As a follow-up work, the Generic Attention Explainability (GAE) (Chefer, Gur, and Wolf 2021a) was introduced as a generalization of T-Attr for explaining Bi-Modal transformers. In contrast to T-Attr and GAE, which are specifically designed for transformers, BEE is a versatile approach that can generate explanations for both CNN and ViT models in a unified manner. More recently, Iterated Integrated Attributions (IIA) (Barkan et al. 2023b) proposed a generalization of IG through an iterated integral. While IIA uses a constant baseline for the integration process, BEE models the baseline as a random tensor, enabling the generation of multiple distinct explanation maps. The stochastic nature of BEE allows for the selection of the optimal explanation map from this diverse set, tailored to the specified metric.

3 Baseline Exploration-Exploitation

Let $\mathbf{x} \in \mathbb{R}^{d_0}$ be the input image. Let $f : \mathbb{R}^{d_0} \to \mathbb{R}^{d_L}$ be a neural network consisting of L layers, each producing a representation $\mathbf{x}^l \in \mathbb{R}^{d_l}$, and $\mathbf{x}^0 := \mathbf{x}$. The final layer produces the prediction $f(\mathbf{x})$, in which the score for the class y is given by $f_y(\mathbf{x})$. Our goal is to produce an explanation map $\mathbf{m} \in \mathbb{R}^{d_0}$ w.r.t. the class y, in which each element $\mathbf{m}_i$ represents the attribution of the prediction $f_y(\mathbf{x})$ to the element $\mathbf{x}_i$. IG (Sundararajan, Taly, and Yan 2017) enables the creation of an explanation map by defining a linear path between a baseline representation $\mathbf{b}$ and $\mathbf{x}$ via the parameterization:

$$\mathbf{v} = (1-a)\mathbf{b} + a\mathbf{x} \text{ with } a \in [0,1], \quad (1)$$

and accumulating the gradients along this path as follows:

$$\mathbf{m}_{IG} = \int_0^1 \frac{\partial f_y(\mathbf{v})}{\partial \mathbf{v}} \circ \frac{\partial \mathbf{v}}{\partial a} da \approx \frac{\mathbf{x}-\mathbf{b}}{n} \circ \sum_{k=1}^{n} \frac{\partial f_y(\mathbf{v})}{\partial \mathbf{v}}, \quad (2)$$

where $\circ$ denotes the elementwise product, and the approximation is obtained by setting $a = \frac{k}{n}$ in Eq. 1. A path between $\mathbf{b}$ and $\mathbf{x}$ symbolizes the transition from the uninformative baseline $\mathbf{b}$, essentially representing missing information, to the informative image $\mathbf{x}$. Therefore, it is crucial to design the baseline representation such that it aligns with the concept of missing information. In Sec. 3.2, we present the types of baselines considered in this work, and the BEE procedure that enables adaptive sampling of baselines. BEE constructs an explanation map by integrating functions that incorporate information from both the internal network representations and their gradients. A distinctive characteristic of BEE is its utilization of stochastic integration paths, wherein the baseline $\mathbf{b}$ is sampled from a distribution $\mathcal{D}$ learned through a contextual EE procedure. This procedure facilitates adaptive baseline sampling from different types of baseline distributions, resulting in the generation of diverse explanation maps. Given the challenge of quantifying the quality of an explanation map, and considering the absence of a universally agreed-upon metric, BEE enables the selection of the best-performing explanation map (or their combination / aggregation) from a diverse set of explanations w.r.t. the specific metric of interest. We start by describing the BEE explanation map generation process, assuming the baseline distribution $\mathcal{D}$ is **given** (Sec. 3.1). Then, in Sec. 3.2, we introduce the BEE procedure that optimizes the baseline distribution $\mathcal{D}$ per metric.

3.1 The BEE explanation map construction

Let $f^l : \mathbb{R}^{d_l} \rightarrow \mathbb{R}^{d_L}$ be a sub-network of f taking an input $\mathbf{x}^l$ and producing the final prediction $f_y^l(\mathbf{x}^l)$. Given a prescribed number of trials T, BEE samples T baselines $\{\mathbf{b}^{lt}\}_{t=1}^{T}$ from the baseline distribution $\mathcal{D}$ and computes a set of explanation maps $M^l = \{\mathbf{m}^{lt}\}_{t=1}^{T}$ as follows:

$$\mathbf{m}^{lt} = u\left(\frac{\mathbf{x}^l - \mathbf{b}^{lt}}{n} \circ \sum_{k=1}^{n} \psi\left(\frac{\partial f_y^l(\mathbf{v}^{lt})}{\partial \mathbf{v}^{lt}}, \mathbf{v}^{lt}\right)\right), \quad (3)$$

where $\mathbf{v}^{lt} = (1 - \frac{k}{n})\mathbf{b}^{lt} + \frac{k}{n}\mathbf{x}^l$ is the interpolated representation, ψ is a function combining information from $\mathbf{v}^{lt}$ and its gradients, and u is a function transforming the resulting explanation map to match the original spatial dimensions of $\mathbf{x}$. The stochastic nature of BEE enables the formation of multiple explanation maps M^l. By considering explanation maps obtained from different network layers, we form a superset of explanation maps $M^{\mathcal{I}} = \cup_{l\in\mathcal{I}} M^l$, where $\mathcal{I}$ is a set of selected layer indexes. Finally, given a metric of interest s that provides an assessment score $s(\mathbf{m})$ for the goodness of the explanation map $\mathbf{m}$, the BEE explanation map is defined by:

$$\mathbf{m}_{\text{BEE}} = \underset{\mathbf{m}\in M^{\mathcal{I}}}{\operatorname{argmax}}\, s(\mathbf{m}). \quad (4)$$

Therefore, $\mathbf{m}_{\text{BEE}}$ is the explanation map that performs the best on the metric s among the maps in $M^{\mathcal{I}}$. It is important to clarify that BEE selects the best-performing explanation from a relatively small set of candidate explanation maps. This set is not guaranteed to include the optimal explanation map.

3.2 Learning the baseline distribution with BEE

In this section, we describe the BEE procedure that produces the (adpative) baseline distribution $\mathcal{D}$ (which was assumed to be given in Sec. 3.1). Previous works have extensively discussed and compared various notions of missingness in the context of attribution baselines (Fong and Vedaldi 2017; Ancona et al. 2017; Kindermans et al. 2019; Sturmfels, Lundberg, and Lee 2020). However, the challenge remains in selecting the appropriate baseline. In this work, we explore different types of baselines: 1) The constant baseline (**Constant**): In (Sundararajan, Taly, and Yan 2017), the authors employed the black baseline. Generally, a value can be drawn from a valid range to form a constant baseline based on this value. Since BEE integrates on the intermediate representation layers in the network (activations), the valid ranges are set based on the minimum and maximum value in each channel of the activation. However, this baseline may not accurately model missingness. For instance, using a constant black image as a baseline in IG may not highlight black pixels as important, even if these pixels constitute the object of interest. 2) The blur baseline (**Blur**): In (Fong and Vedaldi 2017), the authors used a blurred version of the image (or activation in the case of BEE) as a domain-specific technique to represent missing information. This approach is appealing due to its intuitive capture of the concept of missingness in images. The blur operation is controlled by a parameter that determines the kernel size. 3) The uniform baseline (**Uniform**): A potential drawback of the blurred baseline is its bias toward highlighting high-frequency information. Pixels that are very similar to their neighbors may receive less importance than pixels that differ significantly from their neighbors. To address this, missingness can be modeled by sampling a random uniform image (or activation) within the valid pixel range and using it as a baseline. 4) The normal baseline (**Normal**): This baseline is randomly drawn from a Normal distribution centered on the original image (or activation) with a specified variance parameter. 5) The training distribution baseline (**Train Data**): This method involves drawing instances from the training data and using them as baselines to generate multiple explanation maps. These maps are then averaged to produce a final explanation map (similar to Expected Gradients (Erion et al. 2021)). In BEE, each baseline is formed by using the representation produced by the specific layer of interest. Detailed implementation specifics for each baseline type are provided in the Appendix[1].

In (Sturmfels, Lundberg, and Lee 2020), the authors explored the impact of various types of baselines including the aforementioned ones. Additionally, they investigated different baseline aggregation and averaging techniques. However, their findings did not indicate a specific baseline type as the optimal choice. Consequently, we posit that the richer and more comprehensive the baseline distribution $\mathcal{D}$, the higher the probability of sampling the a better-performing baseline. To achieve this goal, we propose to construct $\mathcal{D}$ as a mixture of distributions encompassing multiple types of baselines. In this manner, the process involves first drawing the baseline type according to the mixture weight and subsequently drawing the baseline from the distribution specific to that type.

While a straightforward way would be to set equal

[1]The Appendix appears in the arXiv version of this work.

weights for all mixtures, we propose to learn a distribution for each mixture weight using the BEE approach that employs contextual EE of the baseline: At each iteration, the baseline type is sampled according to the current learned distribution of the mixture weights. This is followed by sampling a baseline of the specific distribution type, generating an explanation map, extracting a reward based on the explanation metric, and updating the distribution of mixture weights accordingly. This process enables the probabilistic selection of the most promising baseline type w.r.t. the specific context (the input $\mathbf{x}$) and explanation metric at hand s. In what follows, we describe this process in detail.

Let $c_\theta(\mathbf{x}) \in \mathbb{R}^K$ represent the context obtained by applying an auxiliary neural network c_θ (parameterized by θ) to $\mathbf{x}$. The role of the function c_θ is to offer information about the particular input $\mathbf{x}$, thereby injecting context into the EE process. This adaptation allows the baseline distribution to adjust based on both the specific input $\mathbf{x}$ and the metric of interest.

Let $\mathcal{B}$ be a set of baseline types (e.g., Constant, Blur, etc.). Each baseline type $b \in \mathcal{B}$ is associated with a random vector $\mathbf{w}^b \in \mathbb{R}^K$ which follows a normal distribution with mean $\mathbf{g}^b$ and diagonal precision matrix represented by a vector $\mathbf{q}^b$. It is important to clarify that $\mathbf{w}^b$ does not represent the distribution of the baseline type b, but rather a random vector serving as a classifier (hyperplane) associated with the specific baseline type b. During the BEE procedure, each $\mathbf{w}^b$ is optimized based on the reward r_b obtained from sampling a baseline of type b. The reward $r_b \in \{1, -1\}$ is modeled as a two-point distribution. Specifically $\Pr(r_b | c_\theta(\mathbf{x}), \mathbf{w}^b) = \sigma(r_b c_\theta(\mathbf{x}) \cdot \mathbf{w}^b)$, where $\sigma(a) = (1 + \exp(-a))^{-1}$ and $\cdot$ denotes the dot-product. Therefore, the dot-product between the context $c_\theta(\mathbf{x})$ and the learned classifier $\mathbf{w}^b$ serves as the (logit) score for the action of sampling a baseline of type b.

This formulation falls within the contextual EE framework: At each step, we play an action, i.e., sampling a baseline type b among $\mathcal{B}$ (under a specific context $c_\theta(\mathbf{x})$), drawing a baseline $\mathbf{b}$ from the selected distribution type[2] b, producing an explanation map $\mathbf{m}$, computing a reward (based on the specific metric of interest), and accordingly updating the relevant set of learnable parameters $\mathbf{g}^b, \mathbf{q}^b$ and θ s.t. the cumulative reward is maximized. It is important to emphasize that the BEE process is employed for each metric separately, enabling optimization per specific metric of interest.

At the beginning of the process, all $\mathbf{g}^b$ and $\mathbf{q}^b$ are set to $\mathbf{0}$ and $\mathbf{1}$, respectively (following the standard normal distribution). Then, at each iteration, given a context $c_\theta(\mathbf{x})$, the following steps are performed:

1. For each $z \in \mathcal{B}$, draw $\mathbf{w}^z$ from a normal distribution (using $\mathbf{g}^z$ and $\mathbf{q}^z$) and set $b \leftarrow \underset{z \in \mathcal{B}}{\operatorname{argmax}}\ \sigma(c_\theta(\mathbf{x}) \cdot \mathbf{w}^z)$.
2. Draw a baseline $\mathbf{b}$ from the baseline distribution of type b, compute an explanation map $\mathbf{m}$ according to Eq. 3 using $\mathbf{b}$ and $\mathbf{x}$.
3. Compute the metric score $s(\mathbf{m})$ and extract a corresponding reward y.
4. $\mathbf{u}^*, \theta^* \leftarrow \underset{\mathbf{u},\theta}{\operatorname{argmin}} -\log \sigma(y\mathbf{u} \cdot c_\theta(\mathbf{x})) + \frac{1}{2}\sum_{i=1}^{K} \mathbf{q}_i^b(\mathbf{u}_i - \mathbf{g}_i^b)^2$.
5. $\mathbf{g}^b \leftarrow \mathbf{u}^*, \theta \leftarrow \theta^*$.
6. $\mathbf{q}_i^b \leftarrow \mathbf{q}_i^b + \sigma(\mathbf{g}^b \cdot c_\theta(\mathbf{x}))\sigma(-\mathbf{g}^b \cdot c_\theta(\mathbf{x}))c_\theta(\mathbf{x})_i^2$.

Step 1 selects the baseline type b with the highest expected reward. Step 2 draws a baseline $\mathbf{b}$ from the selected baseline distribution of type b, and uses it to form an explanation map $\mathbf{m}$. Step 3 produces a reward y based on the metric score. For metrics that output binary explanation scores, the reward is simply mapped to 1 or -1 depending on success / failure. For metrics that outputs continuous scores, we compute the normalized rank $h \in [0, 1]$ of the produced score, which is computed relative to the scores obtained from previous iterations. Then, the reward y is drawn from a two-point distribution variable ($\{1, -1\}$) with a success parameter h. Steps 4-5 solve an optimization problem and update the mean $\mathbf{g}^b$ and the parameters of the context network θ. The first term in the objective in Step 4 is a the likelihood of the reward given the model parameters and the context. The second term is a Gaussian prior serving as a proximal regularization on the update of the mean. The optimization in Step 4 is carried out by gradient descent w.r.t. $\mathbf{u}$ and θ. Finally, the update of the precision $\mathbf{q}^b$ takes place in Step 6 and follows from the Laplace approximation. Once the learning process of $\mathcal{D}$ is completed, drawing a baseline is a straightforward process accomplished by applying Steps 1 and 2 in the above algorithm.

3.3 BEE pretraining and finetuning

In practice, the BEE procedure outlined in Sec. 3.2 can be employed in two distinct phases: a mandatory pretraining phase followed by an **optional** finetuning phase. During the pretraining phase, we utilize a training set comprising instances from either the training or validation dataset. This phase involves training c_θ and pretraining $\mathbf{g}^b$ and $\mathbf{q}^b$ by iteratively applying the BEE procedure described in Sec. 3.2 for each instance in the training set. Specifically, in each epoch, we iterate over the instances in the training set and perform a single update to $\mathbf{g}^b$, $\mathbf{q}^b$, and θ based on the obtained reward. The pretraining phase is conducted offline and culminates in the optimization of the mixture of baseline distributions $\mathcal{D}$. Subsequently, when presented with a test instance, we employ the procedure described in Sec. 3.1 to obtain the most effective explanation map. This involves sampling T baselines from $\mathcal{D}$, computing a pool of T corresponding explanation maps using Eq. 3, and selecting the best-performing explanation map based on the metric of interest, as expressed in Eq. 4. For further enhancement, BEE finetuning can be employed during inference: Given the test instance x, online updates are applied to the pretrained baseline distribution $\mathcal{D}$ by reapplying the BEE procedure, refining $\mathbf{g}^b$ and $\mathbf{q}^b$ specifically for x and the metric at hand (while keeping θ frozen). Therefore, the finetuning phase facilitates ongoing adaptation of the baseline distribution $\mathcal{D}$ to the characteristics of the test instance during the inference process. It is

[2]Note the distinction: $\mathbf{b}$ represents the baseline **representation**, while $b \in \mathcal{B}$ denotes the **type** of the baseline distribution (e.g., Blur, Uniform, etc.) from which $\mathbf{b}$ is sampled.

essential to highlight that the finetuning phase is optional. In the absence of finetuning, $\mathcal{D}$ retains the distribution optimized during the pretraining phase and remains static during inference; that is, $\mathbf{q}^b$ are not subject to further updates. The advantage of maintaining a static distribution lies in the ability to sample multiple baselines in parallel, as $\mathcal{D}$ remains unchanged across samples. Conversely, the finetuning phase necessitates sequential sampling, as $\mathcal{D}$ is refined after each sample based on the extracted reward. In Sec. 4, we comprehensively evaluate BEE in both settings (pretrained and finetuned). Our findings demonstrate that both versions yield state-of-the-art results, with the finetuning phase leading to additional improvements, albeit with an increase in runtime. The complexity of BEE with finetuning depends on the number of samples drawn from the learned baseline distribution. Sequential sampling is required for updates, adding to the overall complexity. In contrast, the complexity of BEE without finetuning is negligible due to the parallelization of the sampling process, and is comparable to the runtime of IG. A detailed theoretical analysis comparing the complexity of BEE to IG is provided in the Appendix.

3.4 BEE implementation for CNN and VIT models

In CNNs, f comprises residual blocks (He et al. 2016), generating 3D tensors representing the activation maps $\mathbf{x}^l$. Additionally, ψ (Eq. 3) is set to the elementwise product, and u averages across the channel axis to obtain a 2D map. The resulting map is then resized via bicubic interpolation to match the spatial dimensions of $\mathbf{x}$. In ViTs (Dosovitskiy et al. 2020), f consists of transformer encoder blocks, each associated with attention matrices. In this work, we opt to interpolate on the attention matrices. Therefore, we overload the notation and treat $\mathbf{x}^l$ as a 3D tensor in which each channel corresponds to an attention matrix in the l-th layer. Once the baseline matrices are drawn, they are further normalized by softmax to conform to probability distributions. ψ is set to the Gradient Rollout (GR) - a variant of Attention Rollout (Abnar and Zuidema 2020), in which the attention matrices are elementwise multiplied by their gradients. Detailed implementation of GR is provided in our Appendix and on our GitHub repository. The output of GR is the first row of a matrix corresponding to the [CLS] token. u processes the output by truncating its initial element, reshaping it into a 14×14 matrix, and resizing to match the spatial dimensions of the input (with bicubic interpolation). Finally, the architecture of the context network c was set to be a clone of the backbone of f, and was finetuned according the BEE procedure from Sec. 3.2. The exact implementation of BEE for both architectures can be found in our GitHub repository.

4 Experiments

Our experiments aim to address the following research questions (RQs): 1) Does the BEE method outperform state-of-the-art methods? 2) Does BEE finetuning improve upon pretraining? 3) Do different metrics favor different explanation maps and baselines? 4) How does the number of sampled baselines T affect BEE performance? 5) How does the performance of adaptive baseline sampling compare to non-adaptive sampling? 6) Does the learned baseline distribution obtained by BEE converge to the best-performing baseline distribution per metric? 7) Does integration on intermediate representation gradients improve upon integration on input gradients? 8) What is the contribution from context modeling in BEE? 9) Can other path-integration methods benefit from BEE? The primary manuscript addresses RQs 1-6 comprehensively. Specifically, RQs 1-2 are addressed in Tabs. 1 and 2, RQ 3 is addressed in Tab. 3 and Fig. 2, and RQs 4-6 are addressed in Fig. 2. Due to space limitations, experiments addressing RQs 7-9, along with additional analyses and ablation studies, are provided in the Appendix.

4.1 Experimental setup

The experiments were conducted on an NVIDIA DGX 8xA100 Server. Our evaluation includes five model architectures: ResNet101 (**RN**) (He et al. 2016), DenseNet201 (**DN**) (Huang, Liu, and Weinberger 2017), ConvNext-Base (**CN**) (Liu et al. 2022), ViT-Base (**ViT-B**) and ViT-Small (**ViT-S**) (Dosovitskiy et al. 2020).

Objective Evaluation Metrics We conducted an extensive objective evaluation using a comprehensive set of explanation metrics to assess the faithfulness of the generated explanations. This faithfulness evaluation reveals the actual elements in the input the model relies on for its prediction. We consider the following set of metrics: the Area Under the Curve (AUC) of Positive (**POS**) and Negative (**NEG**) perturbations tests (Chefer, Gur, and Wolf 2021b), AUC of the Insertion (**INS**) and Deletion (**DEL**) tests (Petsiuk, Das, and Saenko 2018), AUC of the Softmax Information Curve (**SIC**) and Accuracy Information Curve (**AIC**) (Kapishnikov et al. 2019), Average Drop Percentage (**ADP**) and Percentage Increase in Confidence (**PIC**) (Chattopadhay et al. 2018). For POS, DEL, and ADP the lower the better, while for NEG, INS, SIC, AIC, and PIC the higher the better. It is important to clarify that while we report results for each metric according to its standard protocol, we also conducted experiments using various baselines for masking instead of the standard null baseline (a black image). Our findings indicate that the trends in the results remained consistent, regardless of the baseline used for masking. A detailed description of all metrics is provided in the Appendix.

Datasets In accordance with previous works (Kapishnikov et al. 2019, 2021; Xu, Venugopalan, and Sundararajan 2020; Chefer, Gur, and Wolf 2021b) we use the ImageNet (Deng et al. 2009) ILSVRC 2012 (**IN**) validation set as our test set, which contains 50,000 images from 1,000 classes.

Methods We consider a comprehensive set of explanation methods, covering gradient-based approaches, path-integration techniques, as well as gradient-free methods. Specifically, explanations for CNN models are generated by the following methods: Grad-CAM (**GC**) (Selvaraju et al. 2017), Grad-CAM++ (**GC++**) (Chattopadhay et al. 2018), Iterated Integrated Attributions (**IIA**) (Barkan et al. 2023b), FullGrad (**FG**) (Srinivas and Fleuret 2019), Ablation-

	NEG	POS	INS	DEL	ADP	PIC	SIC	AIC
GC	56.41	17.82	48.14	13.97	17.87	36.69	76.91	74.36
GC++	55.20	18.01	47.56	14.17	16.91	36.53	76.44	71.97
LIFT	55.39	17.53	45.39	15.32	18.03	35.95	76.73	72.76
AC	54.98	19.38	47.05	14.23	16.18	35.52	73.36	70.35
IG	45.66	17.24	39.87	13.49	37.52	19.94	54.67	51.92
GIG	43.97	17.68	37.92	14.18	35.28	18.72	55.04	53.38
BIG	42.25	17.44	36.04	13.95	40.85	24.53	56.98	53.36
FG	54.81	18.06	42.68	14.64	21.06	31.59	75.35	71.49
LC	53.52	17.92	46.11	14.31	24.34	35.43	73.93	65.77
IIA	56.29	16.62	48.01	13.18	12.79	42.96	78.52	75.49
pBEE	59.10	13.69	51.15	11.19	11.35	48.22	81.23	78.45
fBEE	**59.38**	**13.47**	**51.73**	**10.42**	**11.09**	**48.86**	**81.51**	**79.21**

Table 1: Results on the IN dataset for the RN backbone: For POS, DEL and ADP, lower is better. For NEG, INS, PIC, SIC and AIC, higher is better. See Sec. 4.2 for details.

	NEG	POS	INS	DEL	ADP	PIC	SIC	AIC
T-Attr	54.16	17.03	48.58	14.20	54.02	13.37	68.59	61.34
GAE	54.61	17.32	48.96	14.37	37.84	23.65	68.35	57.92
IIA	56.01	15.19	49.31	12.89	33.93	26.18	68.92	62.38
pBEE	58.19	12.51	51.13	10.94	29.05	31.01	71.22	65.32
fBEE	**58.35**	**12.17**	**51.36**	**10.76**	**28.43**	**32.14**	**71.45**	**66.81**

Table 2: Results on the IN dataset using ViT-B: For POS, DEL, and ADP, lower is better. For NEG, INS, PIC, SIC, and AIC, higher is better. See Sec. 4.2 for details.

CAM (**AC**) (Desai and Ramaswamy 2020), Layer-CAM (**LC**) (Jiang et al. 2021), LIFT-CAM (**LIFT**) (Jung and Oh 2021), Integrated Gradients (**IG**) (Sundararajan, Taly, and Yan 2017), Guided IG (**GIG**) (Kapishnikov et al. 2021) and Blur IG (**BIG**) (Xu, Venugopalan, and Sundararajan 2020). For ViT models, we consider the following state-of-the-art explanation methods: Transformer Attribution (**T-Attr**) (Chefer, Gur, and Wolf 2021b), Generic Attention Explainability (**GAE**) (Chefer, Gur, and Wolf 2021a) and IIA (which is applicable both for CNN and ViT architectures). Hyperparameters for all methods were configured according to the recommended settings published by the authors. A detailed description of all explanation methods is provided in the Appendix. Finally, our BEE method is evaluated in two modes: finetuned (**fBEE**) and pretrained (**pBEE**). For the pretraining phase, we used a separate training set of 5000 examples taken from the IN training set, avoiding overlap with the validation set used as a test set. Unless stated otherwise, we sampled $T = 8$ baselines per test instance, and $n = 10$ interpolation steps in the integration process (Eq. 3). The integration was employed on the last convolutional / attention layer, i.e., we set $\mathcal{I} = \{L\}$ (Eq. 4). A comparison of various settings of $\mathcal{I}$, including $L - 1$ and $L - 2$ is presented in the Appendix. The dimension of the context representation K was set to match the output dimension of each backbone separately. Optimization in both the pretraining and finetuning phases was carried out using the Adam optimizer. For precise optimization details, please refer to the Appendix and our GitHub repository.

4.2 Results

Tables 1 and 2 present a quantitative comparison of fBEE, pBEE, and other state-of-the-art explanation methods on RN

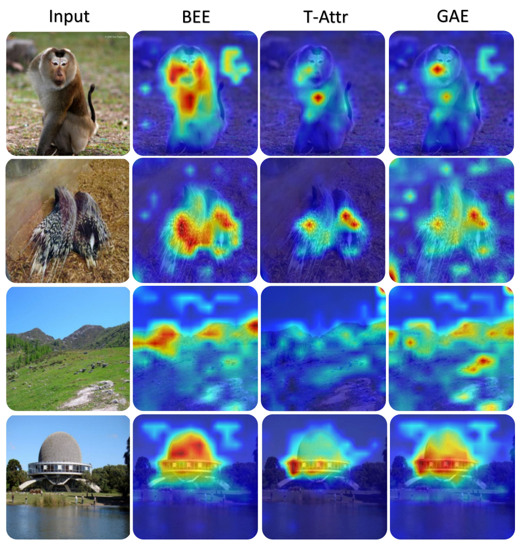

Figure 1: ViT qualitative results: Explanation maps produced using ViT-B w.r.t. the classes (top to bottom): ‘macaque’, ‘porcupine, hedgehog’, ‘alp’ and ‘planetarium’.

and ViT-B models, respectively. The full results, including CN and DN models as well as results for the ViT-S model, are presented in Tables 4 and 5 in Appendix C. The results indicate that fBEE is the top-performing method, with pBEE as the runner-up. The fact both fBEE and pBEE consistently outperform all other methods across all metrics underscores the effectiveness of BEE in explaining vision models. Figure 1 present qualitative comparisons of BEE against other explanation methods for the ViT-B model. Additionally, Figure 6 in the Appendix provides qualitative results for the RN model. Arguably, BEE produces the most focused and class-discriminative explanation maps. These results correlate with the quantitative trends from Tables 1 and 2.

	NEG	POS	INS	DEL	ADP	PIC	SIC	AIC
Normal	**0.46**	0.27	**0.41**	0.19	**0.35**	**0.56**	0.29	0.36
Uniform	0.32	**0.39**	0.26	**0.51**	0.28	0.33	**0.42**	**0.47**
Blur	0.06	0.16	0.09	0.24	0.03	0.08	0.18	0.06
Constant	0.12	0.05	0.13	0.04	0.16	0.02	0.04	0.08
Train Data	0.04	0.13	0.11	0.02	0.18	0.01	0.07	0.03

Table 3: The win-rate distribution (after pretraining) across different metrics and distinct types of baseline distributions.

Table 3 presents the win-rate distribution (after pretraining) for each combination of baseline type and metric, utilizing the RN model. Each column displays the normalized *win-rate* for each of the five baseline types relative to a specific metric. The win-rate was calculated by counting the instances where a particular baseline resulted in the best explanation map for the given metric, followed by normalization. Additionally, Fig.5, in the Appendix, present the reward distribution for each combination of baseline type and metric. The reward distributions were estimated using Monte Carlo approximation to the distribution of $\sigma(c_\theta(\mathbf{x}) \cdot \mathbf{w}^b)$ (by resampling from $\mathbf{w}^b$ for each $b \in \mathcal{B}$). Notably, different metrics exhibit preferences for distinct types of baselines. For instance, PIC, INS, and NEG favor the Normal baseline, while DEL, POS, and SIC favor the Uniform baseline. These analyses demonstrate the need for adaptive adjustment of the baseline, as implemented by BEE.

Finally, we conducted an in-depth analysis of the performance of fBEE and pBEE in comparison to each distinct type of baseline distribution (Normal, Uniform, Train Data, Blur, and Constant), as the number of drawn baselines increases. Using a distinct type of baseline distribution is a special case in which the distribution on the types (mixture weights) conforms to the delta distribution on the specific type, i.e., consistently sampling from the same distinct type of baseline distribution. For completeness, we further consider the opposite extreme, by introducing a non-adaptive baseline sampling strategy (**nBEE**). This strategy involves uniformly drawing a baseline from each of the five distinct baseline distributions. Following the analysis from Tab. 3, we observed that different metrics favor different distinct baselines. For example, the ADP metric favors the Normal baseline. Therefore, if we had an oracle that let us know a priori which is the best baseline type (per combination of input and metric), we could immediately choose it from the beginning. In the following analysis, we aim to investigate the rate of convergence of BEE to the best-performing baseline type, empirically. To this end, each sampling method was executed for 100 iterations, thereby generating 100 baselines. At each iteration, the best-performing baseline (among those sampled so far) was retained, resulting in weakly monotonic graphs. We replicated the experiment with 3000 test examples and reported the mean graph for each method. Figure 2 presents the results for BEE, pBEE, nBEE, and the rest of the distinct types of baseline distribution, using the RN model, focusing on the INS, DEL, ADP, and SIC metrics. A comprehensive figure containing all objective evaluation metrics is available in the Appendix. The horizontal axis in the figure represents the number of samples drawn from the baseline distribution, while the vertical axis corresponds to the metric score. Again, we observe that different metrics favor different baseline types. Notably, fBEE exhibit the fastest convergence to the performance of the best-performing baseline type, thanks to its adaptive nature that promotes the most effective baseline distribution through online updates of the learned mixture of baseline distributions. Interestingly, even though pBEE does not employ finetuning, it marginally underperforms fBEE, indicating that the baseline distribution learned in the pretraining phase is already sufficiently effective on average. In contrast, nBEE, being a non-adaptive method, does not favor any specific baseline type, as it uniformly samples from each type of baseline. Hence, nBEE performs significantly worse, which is expected, as it does not promote the best-performing baseline types and exhausts many samples on less promising baseline types indefinitely. In summary, both fBEE and pBEE demonstrate rapid convergence to the results of the best-performing type of baseline distribution. The findings in Fig. 2 suggest that increased sample size correlates with improved performance. Yet, even with $T = 8$,

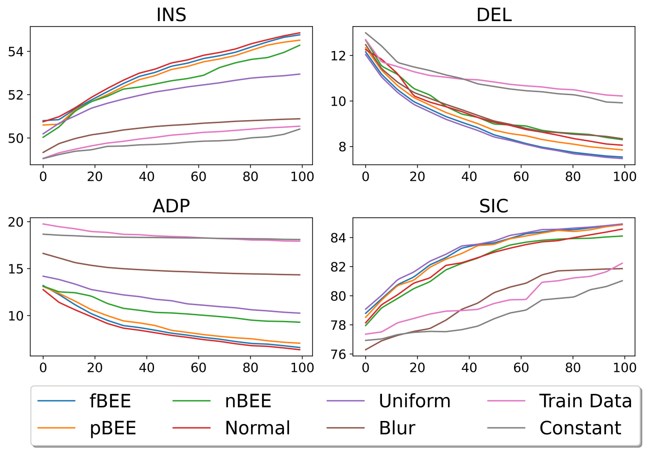

Figure 2: Metric score vs. number of drawn baselines. A comprehensive comparison among fBEE, pBEE, nBEE, and various types of baseline distributions is presented for each metric using the RN model.

both fBEE and pBEE significantly outperforms all other methods, as evident in Tabs. 1 and 2.

Ablation Study Additional ablation studies and analyses can be found in the Appendix. These ablation studies investigate diverse configurations of BEE by varying the number of sampled baselines (T) and the number of interpolation steps (n). Additionally, the analyses in the Appendix underscore the benefits of integrating gradients from intermediate network representations (RQ7), highlight the advantage of incorporating context modeling in BEE (RQ8), and explore the merit of applying the BEE method to other path-integration methods (RQ9), revealing corresponding enhancements in performance.

5 Conclusion

This work recognized the diversity in explanation metrics and baseline representations, highlighting that different metrics exhibit preferences for distinct explanation maps based on the utilization of various baseline types. To address this double diversity, we introduced BEE, a path-integration method that introduces randomness to the integration process by sampling the baseline from a learned mixture of distributions. This mixture is learned through a contextual exploration-exploitation procedure, enhancing performance on the specific metric of interest. BEE can be applied in pretrained (pBEE) and finetuned (fBEE) modes, with the latter continually updating the baseline distribution during inference. Extensive evaluations across various model architectures demonstrate the superior performance of BEE compared to state-of-the-art explanation methods on a variety of objective evaluation metrics. In the Appendix, we further discuss limitations of BEE and avenues for future research.

References

Abnar, S.; and Zuidema, W. 2020. Quantifying Attention Flow in Transformers. *arXiv preprint arXiv:2005.00928*.

Ancona, M.; Ceolini, E.; Öztireli, C.; and Gross, M. 2017. Towards better understanding of gradient-based attribution methods for deep neural networks. *arXiv preprint arXiv:1711.06104*.

Barkan, O.; Asher, Y.; Eshel, A.; Elisha, Y.; and Koenigstein, N. 2023a. Learning to explain: A model-agnostic framework for explaining black box models. In *2023 IEEE International Conference on Data Mining (ICDM)*, 944–949. IEEE.

Barkan, O.; Bogina, V.; Gurevitch, L.; Asher, Y.; and Koenigstein, N. 2024a. A Counterfactual Framework for Learning and Evaluating Explanations for Recommender Systems. In *Proceedings of the ACM on Web Conference 2024*, 3723–3733.

Barkan, O.; Caciularu, A.; Katz, O.; and Koenigstein, N. 2020a. Attentive item2vec: Neural attentive user representations. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 3377–3381. IEEE.

Barkan, O.; Caciularu, A.; Rejwan, I.; Katz, O.; Weill, J.; Malkiel, I.; and Koenigstein, N. 2021a. Representation learning via variational bayesian networks. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 78–88.

Barkan, O.; Elisha, Y.; Asher, Y.; Eshel, A.; and Koenigstein, N. 2023b. Visual Explanations via Iterated Integrated Attributions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2073–2084.

Barkan, O.; Fuchs, Y.; Caciularu, A.; and Koenigstein, N. 2020b. Explainable recommendations via attentive multi-persona collaborative filtering. In *Proceedings of the 14th ACM Conference on Recommender Systems*, 468–473.

Barkan, O.; Hirsch, R.; Katz, O.; Caciularu, A.; and Koenigstein, N. 2021b. Anchor-based collaborative filtering. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2877–2881.

Barkan, O.; Hirsch, R.; Katz, O.; Caciularu, A.; Weill, J.; and Koenigstein, N. 2021c. Cold item integration in deep hybrid recommenders via tunable stochastic gates. In *2021 IEEE International Conference on Data Mining (ICDM)*, 994–999. IEEE.

Barkan, O.; and Koenigstein, N. 2016. Item2vec: neural item embedding for collaborative filtering. In *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*, 1–6. IEEE.

Barkan, O.; Razin, N.; Malkiel, I.; Katz, O.; Caciularu, A.; and Koenigstein, N. 2020c. Scalable attentive sentence pair modeling via distilled sentence embedding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 3235–3242.

Barkan, O.; Rejwan, I.; Caciularu, A.; and Koenigstein, N. 2020d. Bayesian hierarchical words representation learning. In *"Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics"*, 3871–3877.

Barkan, O.; Shvartzman, S.; Uzrad, N.; Elharar, A.; Laufer, M.; and Koenigstein, N. 2023c. InverSynth II: Sound matching via self-supervised synthesizer-proxy and inference-time finetuning. ISMIR.

Barkan, O.; Toib, Y.; Elisha, Y.; Weill, J.; and Koenigstein, N. 2024b. LLM Explainability via Attributive Masking Learning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, 9522–9537.

Barkan, O.; Tsiris, D.; Katz, O.; and Koenigstein, N. 2019. Inversynth: Deep estimation of synthesizer parameter configurations from audio signals. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(12): 2385–2396.

Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; and Zagoruyko, S. 2020. End-to-end object detection with transformers. In *European conference on computer vision*, 213–229. Springer.

Chattopadhay, A.; Sarkar, A.; Howlader, P.; and Balasubramanian, V. N. 2018. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 839–847. IEEE.

Chefer, H.; Gur, S.; and Wolf, L. 2021a. Generic attention-model explainability for interpreting bi-modal and encoder-decoder transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 397–406.

Chefer, H.; Gur, S.; and Wolf, L. 2021b. Transformer interpretability beyond attention visualization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 782–791.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *Computer Vision and Pattern Recognition (CVPR)*.

Desai, S. S.; and Ramaswamy, H. G. 2020. Ablation-CAM: Visual Explanations for Deep Convolutional Network via Gradient-free Localization. *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 972–980.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Elisha, Y.; Barkan, O.; and Koenigstein, N. 2024. Probabilistic Path Integration with Mixture of Baseline Distributions. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, 570–580.

Engel, J.; Hantrakul, L.; Gu, C.; and Roberts, A. 2020. DDSP: Differentiable digital signal processing. *arXiv preprint arXiv:2001.04643*.

Erion, G.; Janizek, J. D.; Sturmfels, P.; Lundberg, S. M.; and Lee, S.-I. 2021. Improving performance of deep learning models with axiomatic attribution priors and expected gradients. *Nature machine intelligence*, 3(7): 620–631.

Fong, R. C.; and Vedaldi, A. 2017. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE International Conference on Computer Vision*, 3429–3437.

Gaiger, K.; Barkan, O.; Tsipory-Samuel, S.; and Koenigstein, N. 2023. Not All Memories Created Equal: Dynamic User Representations for Collaborative Filtering. *IEEE Access*, 11: 34746–34763.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

He, X.; Liao, L.; Zhang, H.; Nie, L.; Hu, X.; and Chua, T.-S. 2017. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, 173–182.

Huang, G.; Liu, Z.; and Weinberger, K. Q. 2017. Densely Connected Convolutional Networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2261–2269.

Jiang, P.-T.; Zhang, C.-B.; Hou, Q.; Cheng, M.-M.; and Wei, Y. 2021. Layercam: Exploring hierarchical class activation maps for localization. *IEEE Transactions on Image Processing*, 30: 5875–5888.

Jung, H.; and Oh, Y. 2021. Towards Better Explanations of Class Activation Mapping. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 1316–1324.

Kapishnikov, A.; Bolukbasi, T.; Viégas, F.; and Terry, M. 2019. Xrai: Better attributions through regions. In *Proceedings of ICCV*, 4948–4957.

Kapishnikov, A.; Venugopalan, S.; Avci, B.; Wedin, B.; Terry, M.; and Bolukbasi, T. 2021. Guided integrated gradients: An adaptive path method for removing noise. In *Proceedings of CVPR*, 5050–5058.

Katz, O.; Barkan, O.; Koenigstein, N.; and Zabari, N. 2022. Learning to ride a buy-cycle: A hyper-convolutional model for next basket repurchase recommendation. In *Proceedings of the 16th ACM Conference on Recommender Systems*, 316–326.

Kindermans, P.-J.; Hooker, S.; Adebayo, J.; Alber, M.; Sch"utt, K. T.; D"ahne, S.; Erhan, D.; and Kim, B. 2019. The (un) reliability of saliency methods. *Explainable AI: Interpreting, explaining and visualizing deep learning*, 267–280.

Kong, Z.; Ping, W.; Huang, J.; Zhao, K.; and Catanzaro, B. 2020. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761*.

Liu, Z.; Mao, H.; Wu, C.; Feichtenhofer, C.; Darrell, T.; and Xie, S. 2022. A ConvNet for the 2020s. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 11966–11976.

Malkiel, I.; Ginzburg, D.; Barkan, O.; Caciularu, A.; Weill, J.; and Koenigstein, N. 2022. Interpreting BERT-based text similarity via activation and saliency maps. In *Proceedings of the ACM Web Conference 2022*, 3259–3268.

Petsiuk, V.; Das, A.; and Saenko, K. 2018. Rise: Randomized input sampling for explanation of black-box models. *arXiv preprint arXiv:1806.07421*.

Selvaraju, R. R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; and Batra, D. 2017. Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, 618–626.

Simonyan, K.; Vedaldi, A.; and Zisserman, A. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.

Srinivas, S.; and Fleuret, F. 2019. Full-Gradient Representation for Neural Network Visualization. In *NeurIPS*.

Sturmfels, P.; Lundberg, S.; and Lee, S.-I. 2020. Visualizing the Impact of Feature Attribution Baselines. *Distill*. Https://distill.pub/2020/attribution-baselines.

Sundararajan, M.; Taly, A.; and Yan, Q. 2017. Axiomatic Attribution for Deep Networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017*, 3319–3328.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

Wang, H.; Wang, Z.; Du, M.; Yang, F.; Zhang, Z.; Ding, S.; Mardziel, P.; and Hu, X. 2020. Score-CAM: Score-Weighted Visual Explanations for Convolutional Neural Networks. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 111–119.

Xu, S.; Venugopalan, S.; and Sundararajan, M. 2020. Attribution in scale and space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9680–9689.

Zeiler, M. D.; and Fergus, R. 2014. Visualizing and understanding convolutional networks. In *European conference on computer vision*, 818–833. Springer.

Zhou, B.; Bau, D.; Oliva, A.; and Torralba, A. 2018. Interpreting deep visual representations via network dissection. *IEEE transactions on pattern analysis and machine intelligence*.

Zhou, B.; Khosla, A.; Lapedriza, A.; Oliva, A.; and Torralba, A. 2016. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2921–2929.