Gradient-based Uncertainty Attribution for Explainable Bayesian Deep Learning

Hanjing Wang
Rensselaer Polytechnic Institute
wangh36@rpi.edu

Dhiraj Joshi
IBM Research
djoshi@us.ibm.com

Shiqiang Wang
IBM Research
wangshiq@us.ibm.com

Qiang Ji
Rensselaer Polytechnic Institute
jiq@rpi.edu

Abstract

Predictions made by deep learning models are prone to data perturbations, adversarial attacks, and out-of-distribution inputs. To build a trusted AI system, it is therefore critical to accurately quantify the prediction uncertainties. While current efforts focus on improving uncertainty quantification accuracy and efficiency, there is a need to identify uncertainty sources and take actions to mitigate their effects on predictions. Therefore, we propose to develop explainable and actionable Bayesian deep learning methods to not only perform accurate uncertainty quantification but also explain the uncertainties, identify their sources, and propose strategies to mitigate the uncertainty impacts. Specifically, we introduce a gradient-based uncertainty attribution method to identify the most problematic regions of the input that contribute to the prediction uncertainty. Compared to existing methods, the proposed UA-Backprop has competitive accuracy, relaxed assumptions, and high efficiency. Moreover, we propose an uncertainty mitigation strategy that leverages the attribution results as attention to further improve the model performance. Both qualitative and quantitative evaluations are conducted to demonstrate the effectiveness of our proposed methods.

1. Introduction

Despite significant progress in many fields, conventional deep learning models cannot effectively quantify their prediction uncertainties, resulting in overconfidence in unknown areas and the inability to detect attacks caused by data perturbations and out-of-distribution inputs. Left unaddressed, this may cause disastrous consequences for safety-critical applications, and lead to untrustworthy AI models.

The predictive uncertainty can be divided into epistemic uncertainty and aleatoric uncertainty [16]. Epistemic un-certainty reflects the model's lack of knowledge about the input. High epistemic uncertainty arises in regions, where there are few or no observations. Aleatoric uncertainty measures the inherent stochasticity in the data. Inputs with high noise are expected to have high aleatoric uncertainty. Conventional deep learning models, such as deterministic classification models that output softmax probabilities, can only estimate the aleatoric uncertainty.

Bayesian deep learning (BDL) offers a principled framework for estimating both aleatoric and epistemic uncertainties. Unlike the traditional point-estimated models, BDL constructs the posterior distribution of model parameters. By sampling predictions from various models derived from the parameter posterior, BDL avoids overfitting and allows for systematic quantification of predictive uncertainties. However, current BDL methods primarily concentrate on enhancing the accuracy and efficiency of uncertainty quantification, while failing to explicate the precise locations of the input data that cause predictive uncertainties and take suitable measures to reduce the effects of uncertainties on model predictions.

Uncertainty attribution (UA) aims to generate an uncertainty map of the input data to identify the most problematic regions that contribute to the prediction uncertainty. It evaluates the contribution of each pixel to the uncertainty, thereby increasing the transparency and interpretability of BDL models. Previous attribution methods are mainly developed for classification attribution (CA) with deterministic neural networks (NNs) to find the contribution of image pixels to the classification score. Unlike UA, directly leveraging the gradient-based CA methods for detecting problematic regions is unreliable. While CA explains the model's classification process, assuming its predictions are confident, UA intends to identify the sources of input imperfections that contribute to the high predictive uncertainties. Moreover, CA methods are often class-discriminative

since the classification score depends on the predicted class. As a result, they often fail to explain the inputs which have wrong predictions with large uncertainty [28]. Also shown by Ancona et al. [1], they are not able to show the troublesome areas of images for complex datasets. Existing CA methods can be categorized into gradient-based methods [15, 31, 33–37, 41, 43] and perturbation-based methods [7, 10, 11, 29, 30, 42]. The former directly utilizes the gradient information as input attribution, while the latter modifies the input and observes the corresponding output change. However, perturbation-based methods often require thousands of forward propagations to attribute one image, suffering from high complexity and attribution performance varies for different chosen perturbations. Although CA methods are not directly applicable, we will discuss their plain extensions for uncertainty attribution in Sec. 2.2.

Recently, some methods are specifically proposed for UA. For example, CLUE [3] and its variants [20, 21] aim at generating a better image with minimal uncertainty by modifying the uncertain input through a generative model, where the attribution map is generated by measuring the difference between the original input and the modified input. Perez et al. [28] further combine CLUE with the path integral for improved pixel-wise attributions. However, these methods are inefficient for real-time applications because they require solving one optimization problem per input for a modified image. Moreover, training generative models is generally hard and can be unreliable for complex tasks.

We propose a novel gradient-based UA method, named UA-Backprop, to effectively address the limitations of existing methods. The contributions are summarized below.

- UA-Backprop backpropagates the uncertainty score to the pixel-wise attributions, without requiring a pretrained generative model or additional optimization. The uncertainty is fully attributed to satisfy the completeness property, i.e., the uncertainty can be decomposed into the sum of individual pixel attributions. The explanations can be generated efficiently within a single backward pass of the BDL model.

- We introduce an uncertainty mitigation approach that employs the produced uncertainty map as an attention mechanism to enhance the model's performance. We present both qualitative and quantitative evaluations to validate the efficacy of our proposed method.

2. Preliminaries

2.1. BDL and Uncertainty Quantification

BDL models assume that the neural network parameters $\boldsymbol{\theta}$ are random variables, with a prior $p(\boldsymbol{\theta})$ and a likelihood $p(\mathcal{D}|\boldsymbol{\theta})$, where $\mathcal{D}$ represents the training data. We can apply the Bayes' rule to compute the posterior of $\boldsymbol{\theta}$, i.e., $p(\boldsymbol{\theta}|\mathcal{D})$

as shown in the following equation:

$$p(\boldsymbol{\theta} \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathcal{D})}. \tag{1}$$

Computing the posterior analytically is often intractable. Therefore, various methods have been proposed for approximately generating parameter samples from the posterior, including MCMC sampling methods [6, 12, 13], variational methods [5, 22–24], and ensemble-based methods [14,19,38–40]. The advantages of the BDL models are their capability to quantify aleatoric and epistemic uncertainties.

Let us denote the input as $\boldsymbol{x}$, the target variable as $\boldsymbol{y}$, and the output target distribution as $p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{\theta})$ parameterized by $\boldsymbol{\theta}$, which are the Bayesian parameters such that $\boldsymbol{\theta} \sim p(\boldsymbol{\theta}|\mathcal{D})$. In this paper, we will focus on classification tasks. For a given input $\boldsymbol{x}$ and training data $\mathcal{D}$, we estimate the epistemic uncertainty and the aleatoric uncertainty by the mutual information and the expected entropy [9] in:

$$\underbrace{\mathcal{H}\left[p(\boldsymbol{y}|\boldsymbol{x},\mathcal{D})\right]}_{\text{Total Uncertainty } U_t} = \underbrace{\mathcal{I}\left[\boldsymbol{y},\boldsymbol{\theta}|\boldsymbol{x},\mathcal{D}\right]}_{\text{Epistemic Uncertainty } U_e} + \underbrace{\mathbb{E}_{p(\boldsymbol{\theta}|\mathcal{D})}\left[\mathcal{H}[p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{\theta})]\right]}_{\text{Aleatoric Uncertainty } U_a} \tag{2}$$

where $\mathcal{H}$, $\mathcal{I}$, and $\mathbb{E}$ represent the entropy, mutual information, and expectation, respectively. Using Monte Carlo approximation of the posterior, we have

$$\mathcal{H}\left[p(\boldsymbol{y}|\boldsymbol{x},\mathcal{D})\right] = \mathcal{H}\left[\mathbb{E}_{p(\boldsymbol{\theta}|D)}[p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{\theta})]\right] \tag{3a}$$

$$\approx \mathcal{H}\left[\frac{1}{S}\sum_{s=1}^{S} p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{\theta}^s)\right] \tag{3b}$$

$$\mathbb{E}_{p(\boldsymbol{\theta}|\mathcal{D})}\left[\mathcal{H}[p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{\theta})]\right] \approx \frac{1}{S}\sum_{s=1}^{S}\mathcal{H}[p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{\theta}^s)] \tag{3c}$$

where $\boldsymbol{\theta}^s \sim p(\boldsymbol{\theta}|\mathcal{D})$ and S is the number of samples.

2.2. Gradient-based Uncertainty Attribution

The gradient-based attribution methods can efficiently generate uncertainty maps via backpropagation. While current CA methods mainly utilize the gradients between the model output and input, some of them can be directly extended for UA by using the gradients from the uncertainty to the input. However, raw gradients can be noisy, necessitating the development of various approaches for smoothing gradients, including Integrated Gradient (IG) [37] with its variants [15, 41], SmoothGrad [34], Grad-cam [31], and FullGrad [36]. Some methods use layer-wise relevance propagation (LRP) to construct classification attributions. Although the LRP-based methods [4, 25, 32] can backpropagate the model outputs layer-wisely to the input, there is no direct extension for the uncertainties since we focus on explaining output variations instead of output values. Moreover, they often require specific NN architectures where the
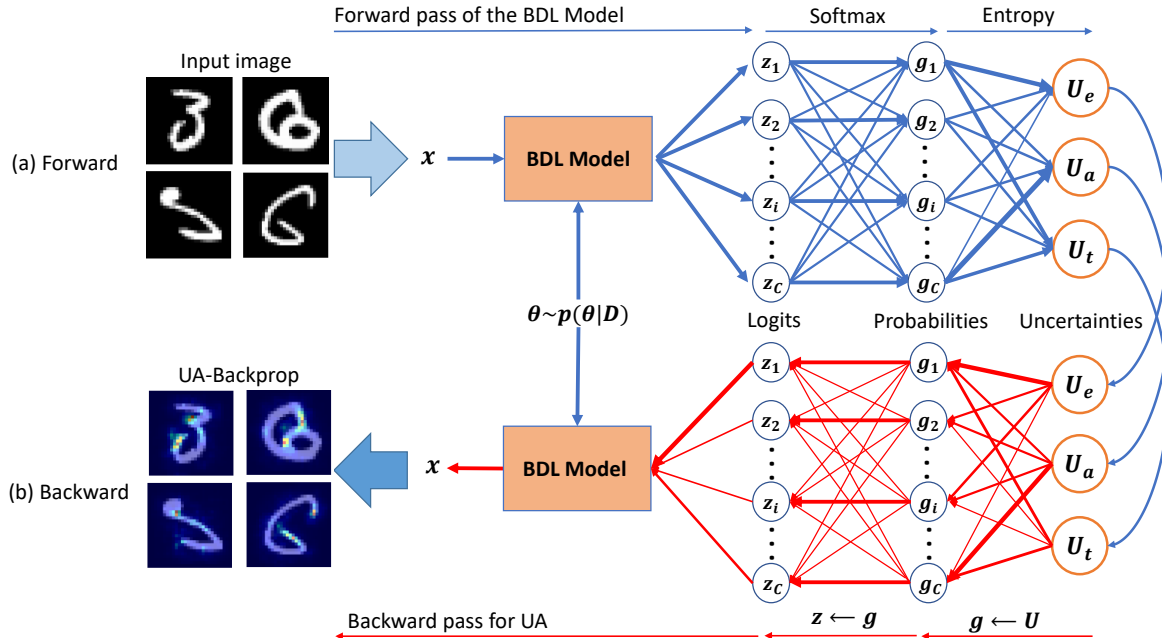
Figure 1. The overall framework of the proposed method. Figure (a) shows the forward propagation of the BDL model for uncertainty quantification. Figure (b) demonstrates the backward process from the uncertainty to the input for attribution analysis, crossing the softmax probabilities, the logits, and the BDL model. The brighter regions indicate higher attributions.

entropy and softmax functions for uncertainty estimation will violate their requirements. In this paper, we consider the vanilla extension of SmoothGrad, FullGrad, and IG-based methods as baselines. Please refer to Appendix A and the survey papers [1, 2, 27] for more discussions.

We contend that the straightforward application of existing attribution methods may not be adequate for conducting UA. Our approach relies on three crucial goals: (1) the uncertainty should be fully attributed with the completeness property satisfied; (2) the pixel-wise attributions should be positive due to data imperfections; (3) the proposed approach should prevent gradient-vanishing issues. Vanilla backpropagation of uncertainty gradients often suffers from the vanishing gradients because of the small magnitude of uncertainty estimates. The resulting visualizations may have "scatter" attributions, which are incomprehensible. Since vanilla adoption of existing methods for deterministic NNs would always violate some of these goals, it is necessary to establish a new gradient-based UA method with competitive accuracy and high efficiency.

3. Uncertainty Attribution with UA-Backprop

3.1. Overall Framework

As shown in Figure 1, let $z(x, \theta) \in \mathcal{R}^C$ denote the output of the neural network with input x parameterized by θ, which is the probability logit before the softmax layer. The number of classes is represented by C. The probability vector $g(x, \theta)$ is generated from $z(x, \theta)$ through

the softmax function, i.e., $g(x, \theta) = \text{softmax}(z(x, \theta))$, where $g_i(x, \theta) = \frac{\exp(z_i(x, \theta))}{\sum_{j=1}^{C} \exp(z_j(x, \theta))}$. For simplicity, we write $z(x, \theta)$ as z and $g(x, \theta)$ as g. Since the complex posterior distribution $p(\theta|\mathcal{D})$ is often intractable, we use a sample-based approximation. We assume that $\{\theta^s\}_{s=1}^S$ are drawn from $p(\theta|\mathcal{D})$, leading to samples $\{z^s\}_{s=1}^S$ and $\{g^s\}_{s=1}^S$. During forward propagation, $\{g^s\}_{s=1}^S$ is used to calculate the epistemic uncertainty U_e, aleatoric uncertainty U_a, and total uncertainty U_t. Let U represent one of the uncertainties in general. For the backpropagation, the uncertainty traverses $U \rightarrow g \rightarrow z \rightarrow x$. The pseudocode for UA-Backprop is provided in Algorithm 1.

Basically, the contribution of each g_i to U, referred to as U_{g_i}, is first computed. Since the backward pass of the BDL model contains S paths $g^s \rightarrow z^s \rightarrow x$ for $\theta^s \sim p(\theta|\mathcal{D})$, we then obtain the contribution of each z_i^s to U, denoted as $U_{z_i^s}$ by exploring all softmax paths $g_j^s \rightarrow z_i^s$ for $j \in [1, \cdots, C]$. Subsequently, $z_i^s \rightarrow x$ is backpropagated. The UA map $M(x)$ is then generated as the pixel-wise contribution to the uncertainty, which aggregates all existing paths $z_i^s \rightarrow x$ with different $s \in [1, \cdots, S]$ and $i \in [1, \cdots, C]$. To fully attribute the uncertainty, the completeness property is enforced on $M(x)$, as shown in Sec. 3.5. The backward steps are elaborated in the following sections.

3.2. Attribution of Softmax Probabilities

In this section, we calculate the attribution of g to uncertainty U. For any i, we denote the contribution of g_i to

Algorithm 1 UA-Backprop + FullGrad

Input: A BDL model $\boldsymbol{\theta} \sim p(\boldsymbol{\theta}|\mathcal{D})$ with sample approximation $\{\boldsymbol{\theta}^s\}_{s=1}^S$; Normalization hyperparameter τ_1, τ_2; The target input $\boldsymbol{x}$ for explanation.

Ouput: The uncertainty attribution map $M(\boldsymbol{x})$.

Step 1 ($U \rightarrow \boldsymbol{g}$): Compute the attribution of softmax probabilities $\{U_{g_i}\}_{j=1}^C$ based on Eq. (4).

Step 2 ($\boldsymbol{g} \rightarrow \boldsymbol{z}$): Based on Eq. (5) and Eq. (8), compute the attribution of each logit $U_{z_i^s}$.

Step 3 ($\boldsymbol{z} \rightarrow \boldsymbol{x}$): Generate the uncertainty attribution map with the aggregation from all paths $z_i^s \rightarrow \boldsymbol{x}$ based on Eq. (9) and Eq. (10).

U_e, U_a, and U_t as U_{e,g_i}, U_{a,g_i}, and U_{t,g_i}, respectively. In general, we denote U_{g_i} as the attribution of g_i to U. By utilizing Eq. (3), we can express U_e, U_a, and U_t in terms of $\{\boldsymbol{g}^s\}_{s=1}^S$, and subsequently decompose them into the sum of individual attributions, as shown in the following equation:

$$U_{t,g_i} = -\left(\frac{1}{S}\sum_{s=1}^S g_i^s\right)\log\left(\frac{1}{S}\sum_{s=1}^S g_i^s\right) \quad (4a)$$

$$U_{a,g_i} = \frac{1}{S}\sum_{s=1}^S -g_i^s \log g_i^s \quad (4b)$$

$$U_{e,g_i} = U_{t,g_i} - U_{a,g_i}, \quad (4c)$$

In general, we can observe that U_{e,g_i}, U_{a,g_i}, U_{t,g_i} only depend on g_i and are independent of other elements of $\boldsymbol{g}$. Moreover, the uncertainties are completely attributed to the softmax probability layer, i.e., $U_t = \sum_{i=1}^C U_{t,g_i}$, $U_a = \sum_{i=1}^C U_{a,g_i}$, $U_e = \sum_{i=1}^C U_{e,g_i}$. When backpropagating the path $\boldsymbol{g}^s \rightarrow \boldsymbol{z}^s$ to get the attribution of logits, U_{g_i} is shared across samples $\{g_i^s\}_{s=1}^S$.

3.3. Attribution of Logits

In this section, we aim to derive U_{e,z_i^s}, U_{a,z_i^s}, and U_{t,z_i^s} as the contribution of z_i^s to U_e, U_a, and U_t by investigating the path from $\boldsymbol{g}^s$ to $\boldsymbol{z}^s$. We introduce $c_{g_j^s \rightarrow z_i^s} \in (0, 1)$ as the coefficient that represents the proportion of the uncertainty attribution that z_i^s receives from g_j^s. Through collecting all the messages from $\{g_j^s\}_{j=1}^C$, the contribution of z_i^s to U, donated as $U_{z_i^s}$, is a weighted combination of the attributions received from the previous layer:

$$U_{z_i^s} = \sum_{j=1}^C c_{g_j^s \rightarrow z_i^s} U_{g_j}. \quad (5)$$

To satisfy the completeness property, it is expected that U_{g_j} is fully propagated into the logit layer as shown in the following equation:

$$U_{g_j} = \sum_{i=1}^C c_{g_j^s \rightarrow z_i^s} U_{g_j}, \quad (6)$$

which is a commonly held assumption in many message-passing mechanisms. Eq. (6) indicates that $\sum_{i=1}^C c_{g_j^s \rightarrow z_i^s} = 1$. In this paper, we apply the softmax gradients to determine $c_{g_j^s \rightarrow z_i^s}$ for the backward step from $\boldsymbol{g}^s$ to $\boldsymbol{z}^s$. Specifically, the gradient of g_j^s to z_i^s is as follows:

$$\frac{\partial g_j^s}{\partial z_i^s} = \begin{cases} g_j^s(1-g_j^s) & \text{if } i = j \\ -g_i^s g_j^s & \text{if } i \neq j \end{cases}. \quad (7)$$

Since $\sum_{k=1}^C g_k^s = 1$ due to the definition of softmax function, it is notable that $|\frac{\partial g_i^s}{\partial z_i^s}| > |\frac{\partial g_j^s}{\partial z_i^s}|$ for $i \neq j$, signifying that g_i^s is the primary source of the attribution for z_i^s. We normalize the gradients to the obtain the coefficients using $\phi(\cdot)$, with the aim of circumventing extremely small coefficients and thus addressing the gradient-vanishing problem. In this study, $\phi(\cdot)$ is a softmax function with temperature τ_1, i.e.,

$$\begin{aligned} c_{g_j^s \rightarrow z_i^s} &= \phi_i\left(\left\{\frac{\partial g_j^s}{\partial z_k^s}\right\}_{k=1}^C, \tau_1\right) \\ &= \frac{\exp\left(\frac{\partial g_j^s}{\partial z_i^s}/(g_j^s \cdot \tau_1)\right)}{\sum_{k=1}^C \exp\left(\frac{\partial g_j^s}{\partial z_k^s}/(g_j^s \cdot \tau_1)\right)}, \end{aligned} \quad (8)$$

where $g_j^s \cdot \tau_1$ is employed for avoiding uniform or extremely small coefficients. It is expected that g_i^s provides the major contribution to z_i^s since the denominator of the softmax function in $\boldsymbol{z}^s \rightarrow \boldsymbol{g}^s$ serves only as a normalization term.

3.4. Attribution of Input

Given the uncertainty attribution $\{U_{z_i^s}\}_{i=1}^C$, associated with $\{z_i^s\}_{i=1}^C$, the attribution map in the input space is generated by backpropagating through $\boldsymbol{z}^s \rightarrow \boldsymbol{x}$. Since each z_i^s may represent different regions of the input, we individually find the corresponding regions of $\boldsymbol{x}$ that contribute to each z_i^s, denoted by $M_i^s(\boldsymbol{x})$. Finally, the uncertainty attribution map $M(\boldsymbol{x})$ is derived by a linear combination of $M_i^s(\boldsymbol{x})$ and $U_{z_i^s}$, i.e.,

$$M(\boldsymbol{x}) = \frac{1}{S}\sum_{s=1}^S \sum_{i=1}^C U_{z_i^s} M_i^s(\boldsymbol{x}). \quad (9)$$

$M(\boldsymbol{x})$ indicates the pixel-wise attributions of U, which is a two-dimensional matrix that has the same height and width as $\boldsymbol{x}$. It is worth noting that during exploring the possible paths for aggregation, the noisy gradients may be smoothed. We notice that some existing gradient-based methods can be used for exploring the path $\boldsymbol{z}^s \rightarrow \boldsymbol{x}$. For example, the magnitude of the raw gradient can be employed such that $M_i^s(\boldsymbol{x}) = |\frac{\partial z_i^s}{\partial \boldsymbol{x}}|$. Especially, more advanced gradient-based methods such as SmoothGrad [34], Grad-cam [31],

and FullGrad [36] can be applied. Intuitively, our proposed method can be a general framework. For the FullGrad method as an example, it aggregates both the gradient of z_i^s with respect to input ($\frac{\partial z_i^s}{\partial \boldsymbol{x}}$) and the gradient of z_i^s with respect to the bias variable $\boldsymbol{b}_l^s$ in each convolutional or fully-connected layer l (i.e., $\frac{\partial z_i^s}{\partial \boldsymbol{b}_l^s}$) to create $M_i^s(\boldsymbol{x})$, i.e.,

$$M_i^s(\boldsymbol{x}) = \psi\left(\left|\frac{\partial z_i^s}{\partial \boldsymbol{x}} \odot \boldsymbol{x}\right| + \sum_l \left|\frac{\partial z_i^s}{\partial \boldsymbol{b}_l^s} \odot \boldsymbol{b}_l^s\right|, \tau_2\right), \quad (10)$$

where $\odot$ is the element-wise product and $|\cdot|$ returns the absolute value. Since different methods will have different scales of $M_i(\boldsymbol{x})$, we apply a post-processing function ψ for normalizing and rescaling the gradients. The function ψ first averages over the channels of $\left|\frac{\partial z_i^s}{\partial \boldsymbol{x}} \odot \boldsymbol{x}\right| +$ $\sum_l \left|\frac{\partial z_i^s}{\partial \boldsymbol{b}_l^s} \odot \boldsymbol{b}_l^s\right|$ and then applies an element-wise softmax function with temperature τ_2. As a general framework, we can leverage the current development of gradient-based attribution methods for deterministic NNs to smooth the gradients and avoid the gradient-vanishing issue.

3.5. Special Properties

Our proposed method satisfies the completeness property, shown in the following equation:

$$U = \sum_{i=1}^{C} U_{g_i} = \sum_{i=1}^{C} U_{z_i^s} = \sum_{(u,v)} M(\boldsymbol{x})[u,v], \quad (11)$$

where (u, v) is the index for the entries of $M(\boldsymbol{x})$. The proof can be found in Appendix A. Our method can also be used with various sensitivity methods for $\boldsymbol{z} \to \boldsymbol{x}$ to satisfy different properties such as implementation invariance and linearity, which are detailed in Appendix A.

4. Uncertainty Mitigation

Leveraging the insights gained from uncertainty attribution, uncertainty mitigation is to develop an uncertainty-driven mitigation strategy to enhance model performance. In particular, the uncertainty attribution map $M(\boldsymbol{x})$ can be utilized as an attention mechanism by multiplying the inputs or features with $1 - M(\boldsymbol{x})$. This can help filter out problematic input information and improve prediction robustness. However, this approach also assigns high weights to unessential background pixels, which is undesirable. To address this issue, the attention weight $A(\boldsymbol{x})$ is defined by the element-wise product of $(1 - M(\boldsymbol{x}))$ and $M(\boldsymbol{x})$ in order to strengthen more informative areas, as shown as follows:

$$A(\boldsymbol{x}) = (1 - M(\boldsymbol{x})) \odot M(\boldsymbol{x}). \quad (12)$$

It is important to note that the attention mechanism can be implemented either in the input space or in the latent space.

In this study, we apply $A(\boldsymbol{x})$ in the latent space, while conducting ablation studies for the input-space attentions in Sec. 5.2.3. Let $\{\boldsymbol{h}_k(\boldsymbol{x})\}_{k=1}^{K}$ with size K be the 2D feature maps generated by the last convolutional layer. We downsample $A(\boldsymbol{x})$ to match the dimensions of $\boldsymbol{h}_k(\boldsymbol{x})$ and utilize $\{(1 + \alpha A(\boldsymbol{x})) \odot \boldsymbol{h}_k(\boldsymbol{x})\}_{k=1}^{K}$ as inputs to the classifier, where α is a hyperparameter that can be tuned. Through retraining using the masked feature maps, the model gains improved accuracy and robustness by ignoring the unimportant background information and the fallacious regions. The complete process is illustrated in Figure 2.
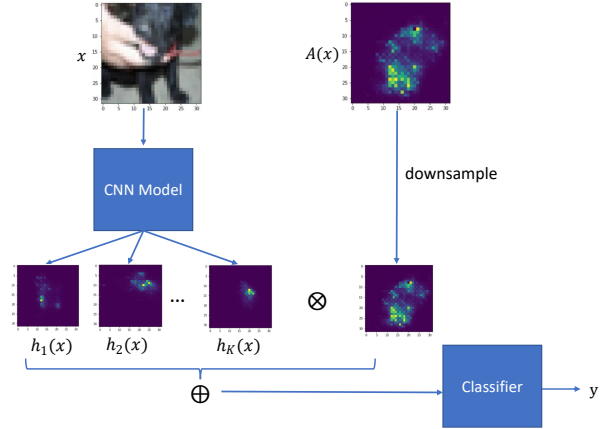

Figure 2. The uncertainty mitigation with attention mechanism.

5. Experiments

Dataset. We evaluate the proposed method on the benchmark image classification datasets including MNIST [8], SVHN [26], CIFAR-10 (C10) [18], and CIFAR-100 (C100) [17].

BDL Model. In our experiments, we use the deep ensemble method [19] for uncertainty quantification, which trains an ensemble of deep neural networks from random initializations. It demonstrates great success in predictive uncertainty calibration and outperforms various approximate Bayesian neural networks [19].

Implementation Details. We use standard CNNs for MNIST/SVHN and Resnet18 for C10/C100. The experiment settings, implementation details, and hyperparameters are provided in Appendix B.

Baselines. We compare our proposed method (UA-Backprop + FullGrad) with various baselines on gradient-based uncertainty attribution. The baselines include the vanilla extension of Grad [33], SmoothGrad [34], FullGrad [36], IG [37], and Blur IG [41] for UA. Although CLUE-variants require a generative model and have low efficiency, we include CLUE [3] and δ-CLUE [20] for comparison.

Evaluation Tasks. In Sec. 5.1, we qualitatively evaluate the UA performance. In Sec. 5.2, we provide the quantitative
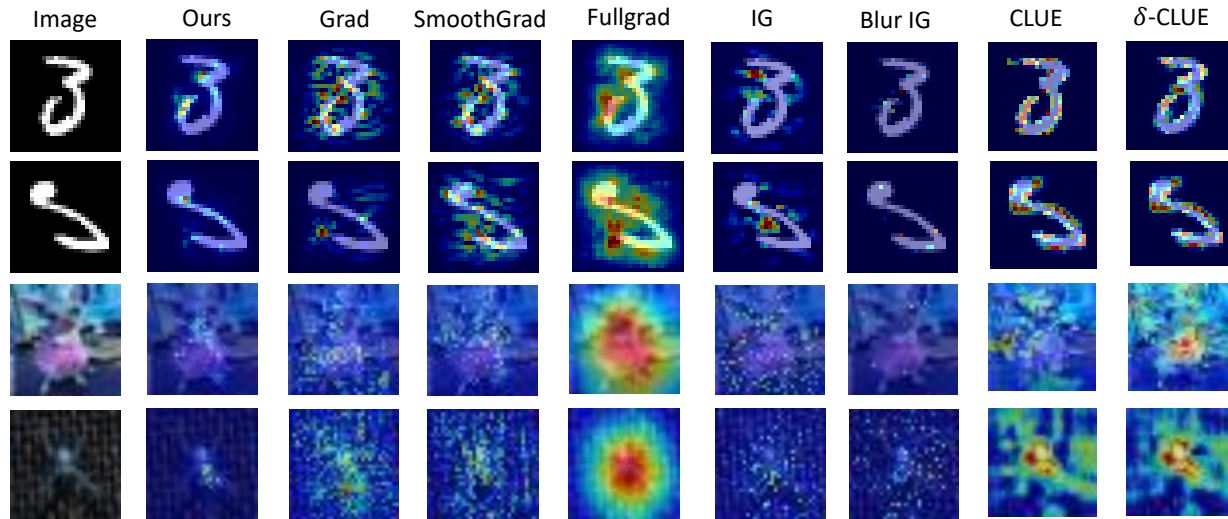
Figure 3. Examples of the epistemic uncertainty attribution maps for various methods on different datasets. Brighter areas indicate essential regions that contribute most to the uncertainty. More examples can be found in Appendix E.

evaluations including the blurring test, and the attention-based uncertainty mitigation. Various supplementary studies are provided in Appendices C and D.

5.1. Qualitative Evaluation

Figure 3 exhibits various examples of attribution maps generated using different techniques. Our analysis reveals that vanilla adoption of CA methods may not be sufficient to generate clear and meaningful visualizations. For instance, as illustrated in Figure 3, we may expect the digit "3" to have a shorter tail, the digit "9" to have a hollow circle with a straight vertical line, and the face of the dog and the small dark body of the spider to be accurately depicted. However, methods such as Grad and Smoothgrad produce ambiguous explanations due to noisy gradients, while FullGrad employs intermediate hidden layers' gradients to identify problematic regions but often lacks detailed information and overemphasizes large central regions. Furthermore, CLUE-based methods tend to identify multiple boundary regions as problematic. They may also fail to provide a comprehensive explanation for complex datasets, where generative models may face significant difficulties in modifying the input to produce an image with lower uncertainty. Finally, CLUE-based methods, Grad, SmoothGrad, and FullGrad fail to fully attribute the uncertainty through the decomposition of pixel-wise contributions. While IG-based methods satisfy the completeness property if the starting image has zero uncertainty, they often produce scattered attributions with minimal regional illustration, posing difficulties in interpretation.

Figure 4 presents various examples of UA maps that depict different types of uncertainties. It is a well-known fact that epistemic uncertainty inversely relates to training data
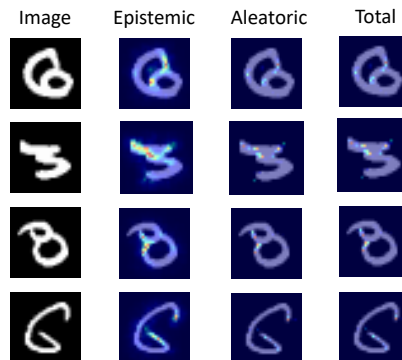

Figure 4. Epistemic, aleatoric, and total uncertainty attribution maps for our proposed method on MNIST dataset.

density. Hence, the epistemic uncertainty maps indicate the areas that deviate from the distribution of training data. In some cases, inserting or blurring pixels will help to reduce uncertainty for performance improvement. The aleatoric uncertainty maps quantify the contribution of input noise to prediction uncertainty, which tends to assign high attributions to object boundaries. As displayed in Figure 4, the total uncertainty maps are quite similar to the aleatoric uncertainty maps. That is because the aleatoric uncertainty quantified in Eq. (2) is often much larger than the epistemic uncertainty, which dominates the total uncertainty.

5.2. Quantitative Evaluation

5.2.1 Blurring Test

Following [28], we evaluate the proposed method through the blurring test. If the most problematic regions are blurred for a highly uncertain image, we expect a significant uncertainty reduction due to the removal of mislead-

Table 1. Attribution performance in terms of MURR and AUC-URR. We evaluate on four different datasets and blur the image with a maximum of 2% or 5% pixels with the highest contribution to the epistemic uncertainty. The bold values indicate the best performance.

Method	Maximum Uncertainty Reduction Rate (MURR) ↑								
	MNIST		C10		C100		SVHN		Avg. Performance
	%2	%5	%2	%5	%2	%5	%2	%5	%2 + %5
Ours	0.648	0.850	0.629	0.848	**0.195**	0.302	0.625	0.758	**0.607**
Grad	0.506	0.741	0.578	0.798	0.165	0.276	0.555	0.705	0.541
SmoothGrad	0.601	0.779	0.566	0.800	0.154	0.255	0.575	0.735	0.558
FullGrad	**0.691**	0.869	0.555	0.772	0.156	0.274	0.565	0.709	0.574
IG	0.434	0.725	0.632	0.827	0.159	0.270	0.649	0.773	0.559
Blur IG	0.305	0.515	**0.693**	**0.971**	0.184	**0.318**	**0.762**	**0.896**	0.581
CLUE	0.614	0.874	0.291	0.628	0.074	0.148	0.171	0.352	0.394
δ-CLUE	0.625	**0.901**	0.415	0.577	0.073	0.150	0.146	0.295	0.398

Method	Area under the Uncertainty Reduction Curve (AUC-URR) ↓								
	MNIST		C10		C100		SVHN		Avg. Performance
	%2	%5	%2	%5	%2	%5	%2	%5	%2 + %5
Ours	0.667	0.445	0.664	0.484	**0.901**	**0.821**	0.526	0.407	**0.614**
Grad	0.709	0.534	0.701	0.538	0.912	0.843	0.613	0.448	0.662
SmoothGrad	0.675	0.461	0.730	0.551	0.919	0.860	0.584	0.424	0.651
FullGrad	**0.603**	0.429	0.696	0.543	0.924	0.859	0.596	0.455	0.638
Blur IG	0.816	0.667	**0.638**	0.466	0.914	0.851	0.541	0.402	0.662
IG	0.752	0.529	0.731	**0.444**	0.905	0.824	**0.523**	**0.298**	0.626
CLUE	0.709	0.397	0.861	0.624	0.966	0.926	0.919	0.815	0.777
δ-CLUE	0.665	**0.395**	0.793	0.710	0.968	0.924	0.932	0.848	0.779

ing information. The blurring can be conducted via a Gaussian filter with mean 0 and standard derivation σ. We iteratively blur the pixels based on their contributions to the uncertainty, where we evaluate the corresponding uncertainty reduction curve to demonstrate the effectiveness of our proposed method. Some examples are shown in Figure 5 and the detailed experiment setting is shown in Appendix B.

The evaluation for the blurring test is conducted on the epistemic uncertainty map since the aleatoric uncertainty captures the input noise and is likely to increase when blurring the image. Denote $v_1, v_2, \cdots, v_T$ as the pixels that contribute most to the epistemic uncertainty, following the decreasing order. We iteratively blur up to t pixels, i.e., $v_{1:t}$, and denote the resulting blurred image as x_t. The uncertainty reduction rate (URR) shown in Eq. (13) quantifies the extent of achieved uncertainty reduction for blurring up to t problematic pixels:

$$\text{URR}(t) = \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} \max_{i \leq t} 1 - \frac{U(x_i)}{U(x)}. \quad (13)$$

For URR, we aggregate the results for various sampled images $x \in \mathcal{X}$. The URR curve, obtained by plotting the decreasing normalized values of $\{\text{URR}(t)\}_{t=1}^{T}$, is a key performance metric. We report two evaluation metrics, namely, the maximum uncertainty reduction rate (MURR), i.e., $\max_{t=1:T} \text{URR}(t)$, and the area under the URR curve (AUC-URR). Larger MURR and smaller AUC-URR values indicate superior performance of the UA method. Since the blurring may lead some images to be out-of-distribution, we report median values instead.

As shown in Table 1, our proposed method achieves the best average performance and ranks among the top three in all datasets. In particular, it consistently outperforms Grad, SmoothGrad, FullGrad, and IG. While Blur IG shows promising performance on certain datasets such as C10 and SVHN, it requires a larger number of blurred pixels to achieve improvements and has no advantages to identify the highest problematic regions. Generative-model-based
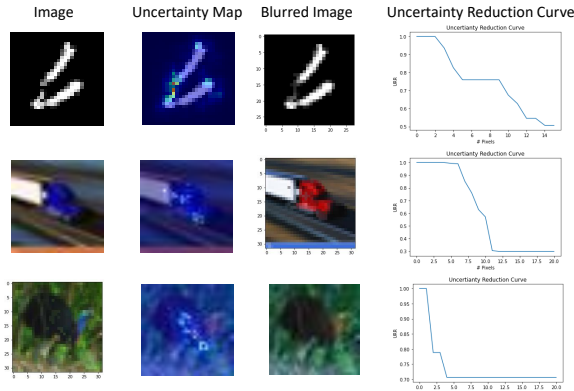

Figure 5. Examples of the blurring test for UA-Backprop.

Table 2. Acc (%) ↑ and NLL ↓ for uncertainty mitigation evaluation. The results are aggregated over 5 independent runs.

Method	MNIST		C10		C100		SVHN		Avg. Performance	
	ACC	NLL	ACC	NLL	ACC	NLL	ACC	NLL	ACC	NLL
Ours	91.95	**0.287**	**36.48**	**1.768**	12.12	4.326	**65.13**	**1.489**	**51.42**	**1.968**
Grad	91.35	0.302	31.60	1.938	12.13	4.422	63.74	1.578	49.71	2.060
SmoothGrad	90.68	0.324	32.05	1.942	**12.57**	4.508	62.35	1.628	49.41	2.100
FullGrad	91.39	0.300	32.85	1.920	12.06	4.574	62.38	1.568	49.67	2.091
IG	**91.98**	0.350	34.43	1.829	11.89	**4.265**	64.31	1.511	50.65	1.989
Blur IG	91.57	0.288	32.20	1.935	12.34	4.630	65.04	1.526	50.29	2.095
CLUE	91.64	0.348	33.34	1.846	12.15	4.299	60.01	1.572	49.29	2.016
δ-CLUE	91.76	0.350	35.02	1.809	12.22	4.362	62.71	1.612	50.43	2.033
No attention	90.78	0.358	31.62	1.921	12.02	4.536	60.64	1.569	48.77	2.096

methods, such as CLUE and δ-CLUE, perform well on MNIST but face difficulties in attributing complex images. Additionally, SmoothGrad, Blur IG, and IG require multiple backward passes to attribute one input, while CLUE and δ-CLUE also require a specific optimization process per image, which makes them less efficient. Overall, our proposed method demonstrates superior performance and stands out as the optimal approach for UA in the blurring test.

5.2.2 Uncertainty Mitigation Evaluation

Building on the methodology in Sec. 4, we adopt pre-generated attribution maps as attention mechanisms to enhance model performance. The formulation of attention, denoted by $A(\boldsymbol{x})$, is presented in Eq. (12), and is exemplified in Figure 6. To ensure consistency in scale across different methods, the attribution map $M(\boldsymbol{x})$ is normalized using the element-wise softmax function before being used in Eq. (12).

The experimental focus is on training with limited data due to the time-consuming process of generating attribution maps for large datasets, particularly for methods such as Blur IG, SmoothGrad, and CLUE. To this end, we randomly select 500, 1000, 2000, and 4000 images from MNIST, C10, SVHN, and C100, respectively. The selected samples are trained with pre-generated attention maps and evaluated on the original testing data. The evaluation metrics used are accuracy (ACC) and negative log-likelihood (NLL). The experimental setup is detailed in Appendix B.

Table 2 presents the results obtained for uncertainty mitigation. The method "no attention" refers to plain training without attention incorporated. Our method demonstrates a 6% improvement in ACC compared to vanilla training, suggesting a promising potential for utilizing attribution maps for further model refinement. Our method consistently outperforms other attribution methods in terms of averaged ACC and NLL. We notice that more significant improvement in NLL often occurs for smaller datasets, whereas C100 is challenging to fit with limited samples, and the performance will be more influenced by stochastic training.
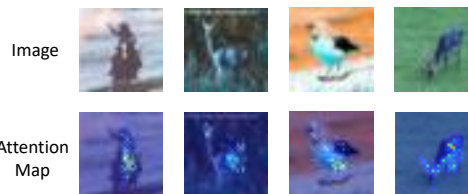

Figure 6. Examples of attention maps for UA-Backprop.

5.2.3 Ablation Studies and Further Analysis

To have a comprehensive evaluation, we conduct the anomaly detection experiment in Appendix C, which compares the predicted problematic regions with the known ground truth. Ablation studies such as efficiency analysis, attribution performances under different experiment settings, and hyperparameter sensitivity analysis are provided in Appendix D.

6. Conclusion

This research aims at developing explainable uncertainty quantification methods for BDL. It will significantly advance the current state of deep learning, allowing it to accurately characterize its uncertainty and improve its performance, facilitating the development of safe, reliable, and trustworthy AI systems. Our proposed method is designed to attribute the uncertainty to the contributions of individual pixels within a single backward pass, resulting in competitive accuracy, relaxed assumptions, and high efficiency. The results of both qualitative and quantitative evaluations suggest that our proposed method has a high potential for producing dependable and comprehensible visualizations and establishing mitigation strategies to reduce uncertainty and improve model performance.

References

[1] Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. *arXiv preprint arXiv:1711.06104*, 2017. 2, 3, 11

[2] Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Gradient-based attribution methods. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, pages 169–191. Springer, 2019. 3, 11

[3] Javier Antorán, Umang Bhatt, Tameem Adel, Adrian Weller, and José Miguel Hernández-Lobato. Getting a clue: A method for explaining uncertainty estimates. *arXiv preprint arXiv:2006.06848*, 2020. 2, 5

[4] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015. 2

[5] Eduardo DC Carvalho, Ronald Clark, Andrea Nicastro, and Paul HJ Kelly. Scalable uncertainty for computer vision with functional variational inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12003–12013, 2020. 2

[6] Tianqi Chen, Emily B. Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, page II–1683–II–1691. JMLR.org, 2014. 2

[7] Piotr Dabkowski and Yarin Gal. Real time image saliency for black box classifiers. *Advances in neural information processing systems*, 30, 2017. 2

[8] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012. 5

[9] Stefan Depeweg, Jose-Miguel Hernandez-Lobato, Finale Doshi-Velez, and Steffen Udluft. Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning. In *International Conference on Machine Learning*, pages 1184–1193. PMLR, 2018. 2

[10] Ruth Fong, Mandela Patrick, and Andrea Vedaldi. Understanding deep networks via extremal perturbations and smooth masks. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2950–2958, 2019. 2

[11] Ruth C Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE international conference on computer vision*, pages 3429–3437, 2017. 2

[12] Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on pattern analysis and machine intelligence*, (6):721–741, 1984. 2

[13] W Keith Hastings. Monte carlo sampling methods using markov chains and their applications. 1970. 2

[14] Gao Huang, Yixuan Li, Geoff Pleiss, Zhuang Liu, John E Hopcroft, and Kilian Q Weinberger. Snapshot ensembles: Train 1, get m for free. *arXiv preprint arXiv:1704.00109*, 2017. 2

[15] Andrei Kapishnikov, Subhashini Venugopalan, Besim Avci, Ben Wedin, Michael Terry, and Tolga Bolukbasi. Guided integrated gradients: An adaptive path method for removing noise. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5050–5058, 2021. 2, 11

[16] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems*, 30, 2017. 1

[17] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 5

[18] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. The cifar-10 dataset. *online: http://www. cs. toronto. edu/kriz/cifar. html*, 55(5), 2014. 5

[19] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6402–6413. Curran Associates, Inc., 2017. 2, 5

[20] Dan Ley, Umang Bhatt, and Adrian Weller. {\delta}-clue: Diverse sets of explanations for uncertainty estimates. *arXiv preprint arXiv:2104.06323*, 2021. 2, 5

[21] Dan Ley, Umang Bhatt, and Adrian Weller. Diverse, global and amortised counterfactual explanations for uncertainty estimates. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 7390–7398, 2022. 2

[22] Christos Louizos and Max Welling. Multiplicative normalizing flows for variational Bayesian neural networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2218–2227, Interna-

tional Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. 2

[23] David J. C. MacKay. A Practical Bayesian Framework for Backpropagation Networks. *Neural Computation*, 4(3):448–472, 1992. 2, 17

[24] Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. e-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 13132–13143. Curran Associates, Inc., 2019. 2

[25] Grégoire Montavon, Sebastian Lapuschkin, Alexander Binder, Wojciech Samek, and Klaus-Robert Müller. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern recognition*, 65:211–222, 2017. 2

[26] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011. 5

[27] Ian E Nielsen, Dimah Dera, Ghulam Rasool, Ravi P Ramachandran, and Nidhal Carla Bouaynaya. Robust explainability: A tutorial on gradient-based attribution methods for deep neural networks. *IEEE Signal Processing Magazine*, 39(4):73–84, 2022. 3, 11

[28] Iker Perez, Piotr Skalski, Alec Barns-Graham, Jason Wong, and David Sutton. Attribution of predictive uncertainties in classification models. In *The 38th Conference on Uncertainty in Artificial Intelligence*, 2022. 2, 6

[29] Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models. *arXiv preprint arXiv:1806.07421*, 2018. 2

[30] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016. 2

[31] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017. 2, 4

[32] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International conference on machine learning*, pages 3145–3153. PMLR, 2017. 2

[33] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013. 2, 5

[34] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017. 2, 4, 5

[35] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014. 2

[36] Suraj Srinivas and François Fleuret. Full-gradient representation for neural network visualization. *Advances in neural information processing systems*, 32, 2019. 2, 5

[37] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017. 2, 5

[38] Matias Valdenegro-Toro. Deep sub-ensembles for fast uncertainty estimation in image classification. *arXiv preprint arXiv:1910.08168*, 2019. 2

[39] Yeming Wen, Dustin Tran, and Jimmy Ba. Batchensemble: an alternative approach to efficient ensemble and lifelong learning. *arXiv preprint arXiv:2002.06715*, 2020. 2

[40] Florian Wenzel, Jasper Snoek, Dustin Tran, and Rodolphe Jenatton. Hyperparameter ensembles for robustness and uncertainty quantification. *arXiv preprint arXiv:2006.13570*, 2020. 2

[41] Shawn Xu, Subhashini Venugopalan, and Mukund Sundararajan. Attribution in scale and space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9680–9689, 2020. 2, 5, 11

[42] Qing Yang, Xia Zhu, Jong-Kae Fwu, Yun Ye, Ganmei You, and Yuan Zhu. Mfpp: Morphological fragmental perturbation pyramid for black-box model explanations. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 1376–1383. IEEE, 2021. 2

[43] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014. 2

A. Properties of UA Methods

In this section, we present an overview of the key characteristics of uncertainty attribution methods, which are extended from the attribution methods for deterministic NNs. We also provide a brief introduction to various existing gradient-based attribution methods for deterministic NNs, along with their vanilla extensions. Lastly, we provide a theoretical demonstration of the essential properties of our proposed method.

A.1. Essential Properties for Uncertainty Attribution

Adopted from the survey papers [1, 2, 27] for attribution methods of deterministic NNs, some important properties are extended for uncertainty attribution of BDL models.

- **Implementation Invariance.** The uncertainty attribution methods should assign the same attribution score to the same input for equivalent neural networks, no matter how they are implemented.

- **Completeness.** The uncertainty score can be fully decomposed into the sum of individual attributions of the input features.

- **Sensitivity.** The attribution methods should assign zero attribution to the features that will not affect the uncertainty. For two inputs that are different in one feature, this feature should be assigned non-zero attribution if the two inputs lead to different uncertainties.

- **Saturation.** Saturation demonstrates a phenomenon in that we assign zero attribution for the regions with zero gradients. The attribution methods should provide tools to avoid saturation.

- **Linearity.** Denote f_1, f_2 as two different BDL models and $M_1(\boldsymbol{x}), M_2(\boldsymbol{x})$ as the corresponding attribution maps for $\boldsymbol{x}$. The linear combination of the two BDL models is $af_1 + bf_2$, where $a, b \in [0, 1]$ and $a + b = 1$. If the linearity is satisfied, the attribution map for $af_1 + bf_2$ is $aM_1(\boldsymbol{x}) + bM_2(\boldsymbol{x})$.

- **Positivity.** Attribution methods should assign non-negative values to input features. Since features are always imperfect, they should positively contribute to the uncertainty unless they are irrelevant.

- **Fidelity.** The features with higher attribution scores should be more sensitive to uncertainty change. Through certain changes in the problematic regions, the uncertainty should be significantly reduced.

A.2. Further Discussion on the Vanilla Extensions of Existing Gradient-based Methods

- **Grad**. For this method, we use the magnitude of the raw gradients from the uncertainty U to the input $\boldsymbol{x}$, shown in Eq. (14):

$$M_G(\boldsymbol{x}) = \left| \frac{\partial U}{\partial \boldsymbol{x}} \right|. \tag{14}$$

- **SmoothGrad**. SmoothGrad tries to smooth the noisy gradients by aggregating from the attributions of various noisy images. Donote K as the number of noisy images we generate through adding Gaussian noises, the attribution map of SmoothGrad is shown in Eq. (15):

$$M_{SG}(\boldsymbol{x}) = \frac{1}{K} \sum_{k=1}^{K} M_G(\boldsymbol{x} + \mathcal{N}(0, \sigma^2 I)) \tag{15}$$

where $\mathcal{N}(0, \sigma^2 I)$ represents the random noise sampled from the Gaussian distribution with 0 mean and covariance matrix $\sigma^2 I$. σ is a hyperparameter and I is the identity matrix.

- **FullGrad**. The FullGrad method calculates the attribution map $M_{FG}(\boldsymbol{x})$ by considering both the gradient of the uncertainty measure U with respect to the input $\boldsymbol{x}$ (i.e., $\frac{\partial U}{\partial \boldsymbol{x}}$) and the gradient of U with respect to the bias variable $\boldsymbol{b}_l$ in every convolutional or fully-connected layer l (i.e., $\frac{\partial U}{\partial \boldsymbol{b}_l}$). This aggregation is mathematically expressed in Eq. (16):

$$M_{FG}(\boldsymbol{x}) = \psi \left(\left| \frac{\partial U}{\partial \boldsymbol{x}} \odot \boldsymbol{x} \right| + \sum_l \left| \frac{\partial U}{\partial \boldsymbol{b}_l} \odot \boldsymbol{b}_l \right| \right) \tag{16}$$

where $\odot$ is the element-wise product and $|\cdot|$ returns the absolute values. ψ is a post-processing function for normalizing and rescaling the gradients.

- **Itegrated Gradient (IG)**. Integrated gradient method creates a path integral from a reference image $\boldsymbol{x}_0$ to $\boldsymbol{x}$, shown in Eq. (17):

$$M_{IG}(\boldsymbol{x}) = (\boldsymbol{x} - \boldsymbol{x}_0) \odot \int_0^1 \frac{\partial U(\boldsymbol{x}_0 + \alpha(\boldsymbol{x} - \boldsymbol{x}_0))}{\partial \boldsymbol{x}} d\alpha. \tag{17}$$

Since IG requires a reference image $\boldsymbol{x}_0$ and the attribution results highly depend on the difference between the reference image and the original image, various extensions are proposed, leading to Blur IG [41] and Guided IG [15].

Based on the survey papers [1, 2, 27], we briefly summarize the properties satisfied by the aforementioned approaches in Table 3. In the next section, we will show the theoretical analysis of our proposed method.

Table 3. The properties of the selected gradient-based attribution methods. The "Yes" in saturation means the attribution method has tools to avoid zero attribution for zero-gradient regions. "*" means the property depends on specific architectures or the chosen layers.

Method	Properties						
	Implementation Invariance	Completeness	Sensitivity	Saturation	Linearity	Positivity	Fidelity
Grad	Yes	No	Yes	No	No	Yes	No
SmoothGrad	Yes	No	Yes	No	No	Yes	Yes
FullGrad	Yes*	Yes	Yes	Yes	No	Yes	Yes
IG	Yes	Yes	Yes	Yes	Yes	No	Yes

A.3. Special Properties of UA-Backprop

Proposition A.1. *UA-Backprop always satisfies the completeness property.*

Proof. Based on Algorithm 1 of the main body of the paper, the uncertainty attribution map generated by our proposed method is shown in Eq. (18):

$$M(\boldsymbol{x}) = \frac{1}{S} \sum_{s=1}^{S} \sum_{i=1}^{C} U_{z_i}^s M_i^s(\boldsymbol{x}) \qquad (18)$$

where $M_i^s(\boldsymbol{x})$ is the normalized relevance map showing the essential regions of $\boldsymbol{x}$ that contribute to $\boldsymbol{z}_i^s$. $U_{z_i}^s$ is the uncertainty attribution of $\boldsymbol{z}_i^s$ received from $\boldsymbol{g}^s$. By taking the sum of $M(\boldsymbol{x})$ over all the elements,

$$\sum_{(u,v)} M(\boldsymbol{x})[u,v]$$

$$= \frac{1}{S} \sum_{s=1}^{S} \sum_{i=1}^{C} U_{z_i}^s \sum_{(u,v)} M_i^s(\boldsymbol{x})[u,v]$$

$$= \frac{1}{S} \sum_{s=1}^{S} \sum_{i=1}^{C} U_{z_i}^s = \frac{1}{S} \sum_{s=1}^{S} \sum_{i=1}^{C} \sum_{j=1}^{C} c_{g_j^s \to z_i^s} U_{g_j} \qquad (19)$$

$$= \frac{1}{S} \sum_{s=1}^{S} \sum_{j=1}^{C} (\sum_{i=1}^{C} c_{g_j^s \to z_i^s}) U_{g_j}$$

$$= \frac{1}{S} \sum_{s=1}^{S} \sum_{j=1}^{C} U_{g_j} = \frac{1}{S} \sum_{s=1}^{S} U = U$$

By incorporating the FullGrad method into the attribution proposed backpropagation framework for the path $\boldsymbol{z} \to \boldsymbol{x}$, our method is able to satisfy several crucial properties. It should be noted that the fulfillment of these properties is primarily contingent on the choice of backpropagation method employed for $\boldsymbol{z} \to \boldsymbol{x}$, as the attribution propagation from $U \to \boldsymbol{g}$ and $\boldsymbol{g} \to \boldsymbol{z}$ does not involve neural network parameters. In the case of UA-Backprop + FullGrad, our method is able to achieve completeness, sensitivity, saturation, positivity, and fidelity.

B. Implementation Details and Experiment Settings

In this section, we will discuss the implementation details of the proposed method and provide further information about the experiment settings.

B.1. Implementation Details and Training Hyperparameters

B.1.1 Model Architecture

As described in Sec. 5 of the main body of the paper, we adopt the deep ensemble method to estimate the uncertainty. Specifically, we train an ensemble of five models for each dataset with different initialization seeds. Common data augmentation techniques, such as random cropping and horizontal flipping, are applied to C10, C100, and SVHN datasets. Our experiments are conducted on an RTX2080Ti GPU using PyTorch. The model architecture and hyperparameters used in our experiments are detailed below.

- **MNIST**. We use the architecture: Conv2D-Relu-Conv2D-Relu-MaxPool2D-Dropout-Dense-Relu-Dropout-Dense-Softmax. Each convolutional layer contains 32 convolution filters with 4×4 kernel size. We use a max-pooling layer with a 2×2 kernel, two dense layers with 128 units, and a dropout probability of 0.5. The batch size is set to 128 and the maximum epoch is 30. We use the SGD optimizer with a learning rate of 0.1 and momentum of 0.9.

- **C10**. For the C10 dataset, we employ ResNet18 as the feature extractor, followed by a single fully-connected layer for classification. We use the stochastic gradient descent (SGD) optimizer with an initial learning rate of 0.1 and momentum of 0.9. The maximum number of epochs is set to 100, and we reduce the learning rate to 0.01, 0.001, and 0.0001 at the 30th, 60th, and 90th epochs, respectively. The batch size is set to 128.

- **C100**. For the C100 dataset, we use the same model architecture as in C10, with ResNet18 as the feature extractor and a single fully-connected layer for classification. We adopt the SGD optimizer with an initial

learning rate of 0.1 and momentum of 0.9. The maximum number of epochs is set to 200, and we decrease the learning rate to 0.01, 0.001, and 0.0001 at the 60th, 120th, and 160th epochs, respectively. The batch size is set to 64.

- **SVHN**. We use the same architecture as MNIST. The batch size is set to 64 and the maximum epoch is 50. We use the SGD optimizer with a learning rate of 0.1 and momentum of 0.9. The learning rate is decreased to 0.01 and 0.001 at the 15th and 30th epochs.

B.1.2 Implementation of the Attribution Approaches

- **Ours**. Regarding the MNIST dataset, we set τ_1 and τ_2 to 0.08 and 0.3 respectively, whereas for C10, C100, and SVHN, we set τ_1 and τ_2 to 0.55 and 0.02. The hyperparameters are different since MNIST contains only grayscale images, while the other datasets consist of colorful images. We utilize the FullGrad method, which is an internal part of the UA-Backprop for $z \rightarrow x$, and we refer to the implementation available at https://github.com/idiap/fullgrad-saliency with the default hyperparameters.

- **Grad**. We use the Torch.autograd to directly compute the gradient from the uncertainty score and the input.

- **SmoothGrad**. Based on Eq. (15), we use $K = 50, \sigma = 0.1$ to smooth the gradients.

- **FullGrad**. We use the implementation in https://github.com/idiap/fullgrad-saliency as a basis and extend it to the uncertainty attribution analysis by computing the full gradients from the uncertainty score to the input. We utilize the default hyperparameters.

- **Blur IG and IG**. We follow https://github.com/Featurespace/uncertainty-attribution for the uncertainty-adapted versions of the Blur IG and IG. The number of path integrations used for Blur IG and IG is set to 100. We use the white starting image for IG.

- **CLUE and δ-CLUE**. For CLUE and δ-CLUE, a two-stage process is performed where we first train two variational autoencoders (VAEs). Specifically, for the MNIST dataset, the VAE implementation follows that of https://github.com/lyeoni/pytorch-mnist-VAE/blob/master/pytorch-mnist-VAE.ipynb. Meanwhile, for C10, C100, and SVHN datasets, we utilize the implementation of https://github.com/SashaMalysheva/Pytorch-VAE, with the same model architectures and the default hyperparameters.

The output layer of the aforementioned implementation is modified to use a sigmoid activation function for the binary cross-entropy loss. Once the VAEs are trained, we apply the CLUE and δ-CLUE methods to learn a modified image for each test data, where the uncertainty loss and the reconstruction loss are weighted equally. We use Adam optimizer with a learning rate of 0.01 and set the maximum iteration to 500 with an early stop criteria based on an L1 patience of $1e - 3$.

B.2. Experiment Settings

B.2.1 Blurring Test

In Sec. 5 of the main context, we examine the performance of the epistemic uncertainty maps in a blurring test. In this test, the key hyperparameter is the standard deviation σ of the Gaussian filter. However, using a fixed σ would be unfair since a small σ would have no impact on the image, while a large σ would cause the blurred images to be out-of-distribution. Different images may require varying degrees of blurriness to reduce uncertainty appropriately. Therefore, we perform an individual search for σ for each image, ensuring that the blurred image has the minimum uncertainty. The search range is from 0 to 20, with a step of 0.2. As our proposed method aims to identify problematic regions by analyzing uncertain images, we focus on the top 500 images with the highest epistemic uncertainty for the blurring test evaluation. Note that for MNIST dataset, only the top 100 uncertain images are selected for evaluation since most of the images have a good quality with low uncertainty. For each metric, the median value is reported considering that some blurred images could be out-of-distribution with increased uncertainty.

B.2.2 Uncertainty Mitigation With Attention Mechanism

In this study, we aim to improve model performance by using pre-generated uncertainty maps as attention to mitigate uncertainty. Following Eq. (12) of the main body of the paper, the uncertainty attribution map $M(\boldsymbol{x})$ is first normalized using an element-wise softmax function and then used for constructing the attention $A(\boldsymbol{x})$. We use bilinear interpolation to rescale $A(\boldsymbol{x})$ to the size of the hidden feature maps. We then do an element-wise product of $(1 + \alpha A(\boldsymbol{x}))$ with the hidden features, where α is a positive real number that controls the strength of the attention. We choose $\alpha = 0.2$ across all datasets and adding 1 is to keep the information of the regions with low importance to ensure no knowledge loss. In the main experiment, we use the epistemic uncertainty maps, while an ablation study for using aleatoric and total uncertainty maps as attention is provided in Appendix D.2.3. To evaluate model robustness, we retrain the model with the attention mechanism under limited data and

Table 4. IoU ↑ and ADA ↑ for anomaly detection for various datasets. The bold values indicate the best performance

Method	C10		C100		SVHN		Avg. Performance	
	IoU	ADA	IoU	ADA	IoU	ADA	IoU	ADA
Ours	**0.353**	**0.285**	**0.363**	**0.375**	**0.217**	**0.124**	**0.311**	**0.261**
Grad	0.141	0.090	0.167	0.135	0.198	0.096	0.169	0.107
SmoothGrad	0.321	0.260	0.316	0.245	0.212	0.114	0.283	0.206
FullGrad	0.341	**0.285**	0.320	0.295	0.206	0.114	0.289	0.231
IG	0.171	0.090	0.170	0.105	0.139	0.052	0.160	0.082
Blur IG	0.182	0.125	0.318	0.290	0.150	0.078	0.217	0.164
CLUE	0.253	0.210	0.208	0.180	0.115	0.042	0.192	0.114
$\delta-$CLUE	0.248	0.240	0.229	0.220	0.105	0.044	0.194	0.168

test on the original testing dataset. With limited data, there is no need for applying complex models. Hence, we use the CNN-based models for all the datasets. The model architecture is Conv2D-Relu-Conv2D-Relu-MaxPool2D-Dropout-Dense-Relu-Dropout-Dense-Relu-Dense-Softmax. Each convolutional layer contains 32 convolution filters with 4×4 kernel size. We use a max-pooling layer with a 2×2 kernel, several dense layers with 128 units, and a dropout probability of 0.5. The maximum training epoch is 120 and the batch size is 128. We use the SGD optimizer with an initial learning rate of 0.1 and momentum of 0.9. The learning rate is decreased at the 30th, 60th, and 90th epoch with a decay rate of 0.2. Additional results for different experiment settings can be found in Appendix D.2.

C. Anomaly Detection

In this section, we employ our method to conduct anomaly detection by leveraging the known ground-truth problematic regions. Specifically, we substitute one patch of each testing image with a random sample from the training data at an identical location. Despite the modified patch still being marginally in-distribution, it mismatches with the remaining regions, creating the ground-truth problematic regions. We perform a quantitative assessment of the efficacy of our proposed method in detecting these anomaly patches.

The experimental evaluation is conducted on three datasets, namely C10, C100, and SVHN. MNIST is excluded from the comparison due to its grayscale nature. To generate the ground-truth problematic regions, we randomly modify a 10 by 10 patch in each testing image by replacing it with a sample from the training data distribution at the same location. Out of the resulting modified images, we select 200 images that exhibit the largest increase in uncertainty compared to the original images, indicating the most problematic areas. Then the epistemic uncertainty maps are generated, based on which, we predict the troublesome regions by fitting a 10 by 10 bounding box that has the highest average attribution score. It is worth noting that we use a brute-force method to identify the predicted 10 by

10 patch. The predicted bounding boxes are compared with the ground-truth counterparts using Intersection over Union (IoU) and anomaly detection accuracy (ADA). The IoU is calculated by dividing the area of the overlap by the area of union, while the detection accuracy is the percentage of images with IoU greater than 0.5.

Figure 7. The anomaly detection examples. The red bounding boxes represent the predicted problematic regions while the orange bounding boxes are the ground truth.

As shown in Figure 7, the predicted problematic bounding boxes are well-matched with the ground truth, indicating the method's capability to accurately identify anomalous regions. The quantitative evaluation in Table 4 reveals that UA-Backprop outperforms other baselines, especially for Grad, IG, Blur IG, CLUE, and δ-CLUE. These baselines perform poorly in detecting anomalous regions, which may be attributed to their limited ability to identify continuous problematic regions (i.e., the 10 by 10 patches), as they tend to detect only scattered locations.

D. Ablation Studies and Further Analysis

D.1. Efficiency Evaluation

In this section, we present a theoretical efficiency analysis of gradient-based methods for generating uncertainty maps. We define the runtime of a single backpropagation as $O(1)$. Our proposed method, along with Grad and FullGrad, can generate the maps within a single backpass, resulting in a runtime of $O(1)$. However, SmoothGrad, IG,

Table 5. Acc (%) and NLL for uncertainty mitigation evaluation of varying number of training samples N on MNIST and C10 datasets. The results are aggregated over 5 independent runs.

Method	MNIST											
	$N = 200$		$N = 500$		$N = 1000$		$N = 1500$		$N = 2000$		Avg. Performance	
	ACC	NLL	ACC	NLL	ACC	NLL	ACC	NLL	ACC	NLL	ACC	NLL
Ours	**85.86**	**0.461**	91.95	**0.287**	**95.65**	0.186	**96.43**	0.161	**96.72**	0.152	**93.32**	**0.249**
Grad	85.02	0.490	91.35	0.302	94.89	0.192	95.76	0.176	96.47	0.159	92.70	0.264
SmoothGrad	85.38	0.480	90.68	0.324	95.15	0.188	95.97	0.171	96.35	0.159	92.71	0.264
FullGrad	84.75	0.503	91.39	0.300	95.23	**0.175**	95.98	**0.153**	96.44	**0.142**	92.76	0.255
IG	82.66	0.563	**91.98**	0.350	94.94	0.220	95.71	0.190	96.34	0.162	92.33	0.297
Blur IG	85.34	0.485	91.57	0.288	95.04	0.184	96.02	0.155	96.48	0.145	92.89	0.252
Non-attention	84.64	0.524	90.78	0.358	95.01	0.221	95.94	0.189	96.29	0.172	92.55	0.293
Method	C10											
	$N = 1000$		$N = 2000$		$N = 3000$		$N = 4000$		$N = 5000$		Avg. Performance	
	ACC	NLL	ACC	NLL	ACC	NLL	ACC	NLL	ACC	NLL	ACC	NLL
Ours	**36.48**	**1.768**	**49.25**	**1.454**	52.17	1.377	**56.92**	**1.255**	57.64	1.222	**50.49**	**1.415**
Grad	31.47	1.945	47.35	1.472	51.62	1.374	46.91	1.508	52.85	1.322	46.04	1.524
SmoothGrad	31.73	1.944	42.72	1.943	48.15	2.482	47.96	2.342	48.02	2.435	43.72	2.229
FullGrad	32.53	1.920	46.67	1.485	51.46	1.371	54.57	1.290	53.70	1.297	47.79	1.473
IG	34.43	1.829	47.96	1.472	**53.32**	**1.349**	56.37	1.263	**58.41**	**1.200**	50.10	1.423
Blur IG	31.96	1.932	46.38	1,495	52.11	1.364	52.48	1.335	54.71	1.277	47.53	1.481
Non-attention	31.58	1.922	46.57	1.490	51.25	1.378	54.89	1.281	55.11	1.265	47.88	1.467

and Blur IG require multiple backward passes for attribution analysis, with runtimes of $O(T)$ where T represents the number of backward iterations. For SmoothGrad, the value of T depends on the number of noisy images used for aggregation, while for the IG-based method, T is based on the number of samples generated to approximate the path integral. The CLUE-based methods necessitate solving an optimization problem per input to obtain a modified image for reference, which further extends their runtimes. In Table 6, we provide the empirical results on the runtime required for each baseline to attribute a single image, demonstrating that our proposed method outperforms various baselines in terms of computational efficiency.

Table 6. Runtime (s) for attributing one image.

Dataset/Method	Ours	Blur-IG	SmoothGrad	CLUE
MNIST	**0.34**	3.39	3.06	6.93
C10	**0.46**	4.06	3.59	18.43

D.2. Different Experiment Settings for Uncertainty Attention Mitigation

D.2.1 Varying Number of Training Samples

In this section, we present the results of a study in which we investigate the effect of varying the number of training samples on the MNIST and C10 datasets in the context of the retaining with the attention mechanism. The experimental outcomes are reported in Table 5. We observe that our proposed method consistently outperforms other methods

when the training data is limited, as evidenced by the improved testing accuracy and NLL. We also find that adding attention to the training of the C10 dataset may not be beneficial for some methods, possibly due to the noisy gradients.

D.2.2 Varying Hyperparameters

In this section, we investigate the impact of the attention weight coefficient, denoted by α, on the performance of our proposed method for MNIST and C10 datasets. We vary α from 0 to 2 with a step of 0.2 and present the results in Table 7. Our proposed method consistently outperforms the plain training without attention ($\alpha = 0$) as we vary α. In this study, we set α to a minimum value of 0.2. Remarkably, even a small value of α leads to a significant improvement. Furthermore, larger values of α progressively accentuate the informative regions, resulting in better performance, as evidenced by the improved results for $\alpha = 1.8, 2$ on MNIST and $\alpha = 1.2, 1.4$ on C10. Considering the stochastic nature of the training process, we note that the model's performance is insensitive to α within a certain range.

D.2.3 Aleatoric and Total Uncertainty Map

In this study, we explore the use of alternative uncertainty maps, namely aleatoric and total uncertainty maps, in place of epistemic uncertainty maps as the attention mechanism. Table 8 presents a comparison of model performance using different types of uncertainty maps. While all maps exhibit a similar accuracy on the MNIST dataset, utilizing the epistemic uncertainty maps results in better fitting

Table 7. Acc (%) and NLL for uncertainty mitigation evaluation of varying α on MNIST and C10 datasets for our proposed method. We randomly select 500 and 1000 training samples for MNIST and C10, respectively. The results are aggregated over 5 independent runs. $\alpha = 0.2$ is used for the main body of the paper.

| $\alpha =$ | Dataset | | | | | |
| | MNIST | | C10 | | Avg. Performance | |
	ACC	NLL	ACC	NLL	ACC	NLL
0.0	90.78	0.358	31.62	1.921	61.20	1.140
0.2	91.95	0.287	36.48	1.768	64.22	1.028
0.4	91.62	0.329	35.22	1.806	63.42	1.068
0.6	91.98	0.320	35.73	1.793	63.86	1.057
0.8	92.07	0.297	**38.39**	**1.735**	65.23	1.016
1.0	92.17	0.299	36.42	1.779	64.30	1.068
1.2	92.28	0.285	37.59	1.750	**64.94**	1.018
1.4	91.86	0.307	38.00	1.737	64.93	1.022
1.6	91.99	0.295	36.24	1.782	64.12	1.038
1.8	**92.52**	**0.269**	36.52	1.760	64.52	1.015
2.0	92.51	**0.269**	37.77	1.743	65.14	**1.006**

Table 8. Acc (%) and NLL for uncertainty mitigation evaluation with different kinds of uncertainty maps.

| Uncertainty | Dataset | | | |
| | MNIST | | C10 | |
	ACC	NLL	ACC	NLL
Epistemic	**91.95**	**0.287**	36.48	1.768
Aleatoric	91.94	0.315	**37.38**	**1.761**
Total	91.60	0.330	35.14	1.810

based on NLL. On the C10 dataset, the aleatoric uncertainty maps yield slightly better performance in both ACC and NLL. Since aleatoric uncertainty captures input noise, the aleatoric uncertainty maps can strengthen the regions with less noise and may benefit when the input imperfections result mainly from input noise. The superior results for aleatoric uncertainty maps on the C10 dataset may be due to the fact that the C10 dataset is noisier than the MNIST dataset.

D.2.4 Input/Latent-space Attention for Uncertainty Mitigation

Table 9 presents our experimental results using UA maps as input-space attention. The weighted inputs $A(\boldsymbol{x}) \odot \boldsymbol{x}$ are obtained by using $A(\boldsymbol{x})$ as input attention. We then use the weighted inputs to retrain the model under the same experimental conditions as described in Appendix B.2.2. Our results demonstrate that using UA maps as input-space attention yields similar performance compared to the results obtained through latent-space experiments.

D.3. Hyperparameter Sensitivity of Our Proposed Method

The temperatures τ_1 and τ_2 used in the normalization functions are crucial hyperparameters in our proposed method. It is necessary to perform normalization in the intermediate steps to ensure the satisfaction of the complete-

ness property. By choosing appropriate values for τ_1 and τ_2, we aim to avoid uniform or overly sharp coefficients. It is essential to avoid setting τ_1 and τ_2 too small or too large, as this would result in uniform or extreme scores. In this section, we show some blurring test results for SVHN, C10, and C100 datasets to evaluate the sensitivity of τ_1, τ_2 within a certain range. In Table 10, the first row shows the hyperparameters used for the experiments of the main body of the paper. We can observe that the performance varies slightly by choosing different hyperparameters within certain ranges. During experiments, we tune τ_1, τ_2 on C10 dataset and use the same hyperparameters ($\tau_1 = 0.55, \tau_2 = 0.02$) for all other datasets with color images. Since MNIST contains only gray-scale images, we use a different set of hyperparameters, i.e., $\tau_1 = 0.08, \tau_2 = 0.3$. It is worth noting that the cross-dataset results are insensitive to the variations of τ_1, τ_2 within certain ranges. Tuning different τ_1, τ_2 for different datasets can further improve the performance.

D.4. Different Methods for the Path $z \to x$

As described in Sec. 3 of the main paper, the UA-Backprop method has the potential to serve as a general framework for utilizing advanced gradient-based techniques to investigate the path from $\boldsymbol{z}$ to $\boldsymbol{x}$. By exploring the path $z_i^s \to \boldsymbol{x}$, we obtain the relevance map $M_i^s(\boldsymbol{x})$, which highlights the crucial regions of $\boldsymbol{x}$ for z_i^s, as presented in Eq. (10) of the main paper. Although we use the Full-Grad method as our primary approach, other gradient-based

Table 9. Mitigation results (ACC ↑,NLL ↓) for MNIST and C10. The comparison is conducted for input-space attention and latent-space attention for uncertainty mitigation.

Method	MNIST		C10		Average	
	ACC	NLL	ACC	NLL	ACC	NLL
Ours-latent	**0.920**	0.287	0.365	1.768	0.642	1.028
Ours-input	0.919	**0.284**	**0.376**	**1.742**	**0.648**	**1.013**

Table 10. MURR and AUC-URR (AUC) of the blurring test for our proposed method with different hyperparameters. The number of blurring pixels is 2% or 5% of the total pixels. The first row shows the hyperparameters used for displaying the main results. The studies are conducted on SVHN dataset.

Hyperparameter	Dataset - SVNH				Dataset - C10				Dataset - C100			
	2%		5%		2%		5%		2%		5%	
	MURR	AUC	MURR	AUC	MURR	AUC	MURR	AUC	MURR	AUC	MURR	AUC
$\tau_1 = 0.55, \tau_2 = 0.02$	0.625	0.526	0.758	0.407	**0.629**	0.664	0.848	0.484	**0.195**	0.901	0.302	0.821
$\tau_1 = 0.50, \tau_2 = 0.02$	0.603	0.550	0.739	0.419	0.622	0.664	0.848	0.496	0.194	**0.900**	**0.304**	0.821
$\tau_1 = 0.60, \tau_2 = 0.02$	0.607	0.540	0.732	0.407	0.626	0.666	0.850	0.489	0.194	0.901	0.303	0.821
$\tau_1 = 0.65, \tau_2 = 0.01$	**0.645**	0.518	**0.771**	0.397	0.617	0.666	0.848	0.491	0.194	0.901	0.298	**0.820**
$\tau_1 = 0.70, \tau_2 = 0.02$	0.595	0.545	0.747	0.419	0.624	**0.660**	**0.854**	**0.480**	0.194	0.901	0.302	0.821
$\tau_1 = 0.55, \tau_2 = 0.03$	0.635	**0.509**	0.758	**0.387**	0.603	0.690	0.848	0.510	0.194	0.905	0.296	0.831
$\tau_1 = 0.55, \tau_2 = 0.04$	0.608	0.562	0.761	0.406	0.592	0.682	0.829	0.508	0.190	0.903	0.294	0.835

techniques can also be employed within the UA-Backprop framework. As a simple baseline, UA-Backprop + Grad uses

$$M_i^s(\boldsymbol{x}) = \psi\left(\frac{\partial z_i^s}{\partial \boldsymbol{x}}\right) \tag{20}$$

where ψ is a softmax function with temperature τ_2, similar to UA-Backprop + FullGrad. However, the raw gradients could be noisy, and advanced gradient-based methods could be used. For example, UA-Backprop + InputGrad uses

$$M_i^s(\boldsymbol{x}) = \psi\left(\boldsymbol{x} \odot \frac{\partial z_i^s}{\partial \boldsymbol{x}}\right) \tag{21}$$

where the input image is used to smooth the gradients. We can also use the integrated gradient (IG) method, which is an extension of InputGrad by creating a path integral from a reference image $\boldsymbol{x}_0$ to input $\boldsymbol{x}$. For UA-Backprop + IG,

$$\begin{aligned} &M_i^s(\boldsymbol{x}) \\ &= \psi\left((\boldsymbol{x}-\boldsymbol{x}_0) \odot \int_0^1 \frac{\partial z_i(\boldsymbol{x}_0 + \alpha(\boldsymbol{x}-\boldsymbol{x}_0), \boldsymbol{\theta}^s)}{\partial \boldsymbol{x}} d\alpha\right) \end{aligned} \tag{22}$$

where $\boldsymbol{x}_0$ could be a black or white image as the reference. In this section, we provide an ablation study of using different gradient-based methods for the path $\boldsymbol{z} \to \boldsymbol{x}$. The blurring test evaluations are provided in Table 11 for MNIST and SVHN datasets. The first row "UA-Backprop + FullGrad" represents the method shown in the main body of the paper. By using other methods for the path $\boldsymbol{z} \to \boldsymbol{x}$, we can also achieve considerable results. For example, UA-Backprop + InputGrad can achieve some improvements for the MNIST dataset. In short, our proposed method can be

a general framework combining the recent development of other gradient-based methods for deterministic NNs.

D.5. UA-Backprop for A Deterministic NN

Our method can be applied to Ensemble-1 where the uncertainty is calculated by the entropy, i.e., the aleatoric uncertainty. However, the results shown in Table 12 are not good due to inadequate uncertainty quantification (UQ). By using more advanced single-network UQ methods, i.e. Laplacian approximation (LA) [23], our UA method can yield improved results. Note that LA can also provide parameter samples from the posterior distribution, which can be directly used for UA-Backprop. Further studies on effectively performing UA on deterministic models can be our future direction. We will also concentrate on developing an end-to-end training approach that produces the attribution maps for a single network during training iterations and integrates the knowledge of UA for further model enhancement.

D.6. Compare to Random Map

In this section, we compare our proposed method with the random map to better illustrate the effectiveness of the proposed method. The random map is generated by sampling each element from the uniform distribution $U[0, 1]$. To this end, the blurring test results are presented in Table 13 to compare the performance of the proposed method against the random maps. The experimental results show that the random maps fail to reduce the uncertainty during the blurring test.

Table 11. MURR and AUC-URR (AUC) of the blurring test for our proposed method with different approachs for $z \rightarrow x$. The number of blurring pixels is 2% or 5% of the total pixels. The studies are conducted on MNIST and SVHN datasets.

| Method | MNIST | | | | SVHN | | | |
| | 2% | | 5% | | 2% | | 5% | |
	MURR	AUC	MURR	AUC	MURR	AUC	MURR	AUC
UA-Backprop + FullGrad	0.648	0.667	**0.850**	0.445	**0.625**	**0.526**	**0.758**	**0.407**
UA-Backprop + Grad	0.519	0.714	0.720	0.532	0.611	0.543	0.712	0.451
UA-Backprop + InputGrad	**0.673**	**0.618**	0.826	**0.413**	0.549	0.598	0.702	0.445
UA-Backprop + IG	0.611	0.641	0.795	0.439	0.529	0.618	0.703	0.456

Table 12. Attribution results (MURR ↑, AUC-URR ↓).

| Method | MNIST (%2) | | C10 (%2) | |
	MURR	AUC-URR	MURR	AUC-URR
Ours-Ensemble-5	**0.648**	**0.667**	**0.629**	**0.664**
Ours-Ensemble-1	0.425	0.828	0.506	0.710
Ours-LA	0.487	0.768	0.534	0.692

Table 13. MURR and AUC-URR (AUC) of the blurring test to compare our proposed method with randomly generated maps. The number of blurring pixels is 2% of the total pixels. The studies are conducted on MNIST and SVHN datasets.

| Method | Dataset | | | |
| | MNIST (2%) | | SVHN (2%) | |
	MURR	AUC	MURR	AUC
Ours	**0.648**	**0.667**	**0.625**	**0.526**
Random	0.023	0.987	0.011	0.992

D.7. Compare to UA-Backprop without Normalization

The normalization steps are required to achieve the completeness property. Nevertheless, an ablation study shows that with normalization, the MURR (2% / 5%) is **0.648/0.850** for MNIST and **0.629/0.848** for C10; without normalization, it can only achieve 0.471/0.797 for MNIST and 0.518/0.727 for C10.

E. Additional Examples

Figure 8 displays supplementary instances of the uncertainty attribution maps generated by various methods across multiple datasets. Our proposed method offers a more understandable and clear visualization of the generated maps compared to the vanilla application of existing CA methods. The latter often yields ambiguous explanations because of the presence of noisy gradients. In contrast, our approach provides a decomposition of pixel-wise contributions that efficiently explains the uncertainty while offering better regional illustrations that could be comprehended by individuals without expertise in the field.
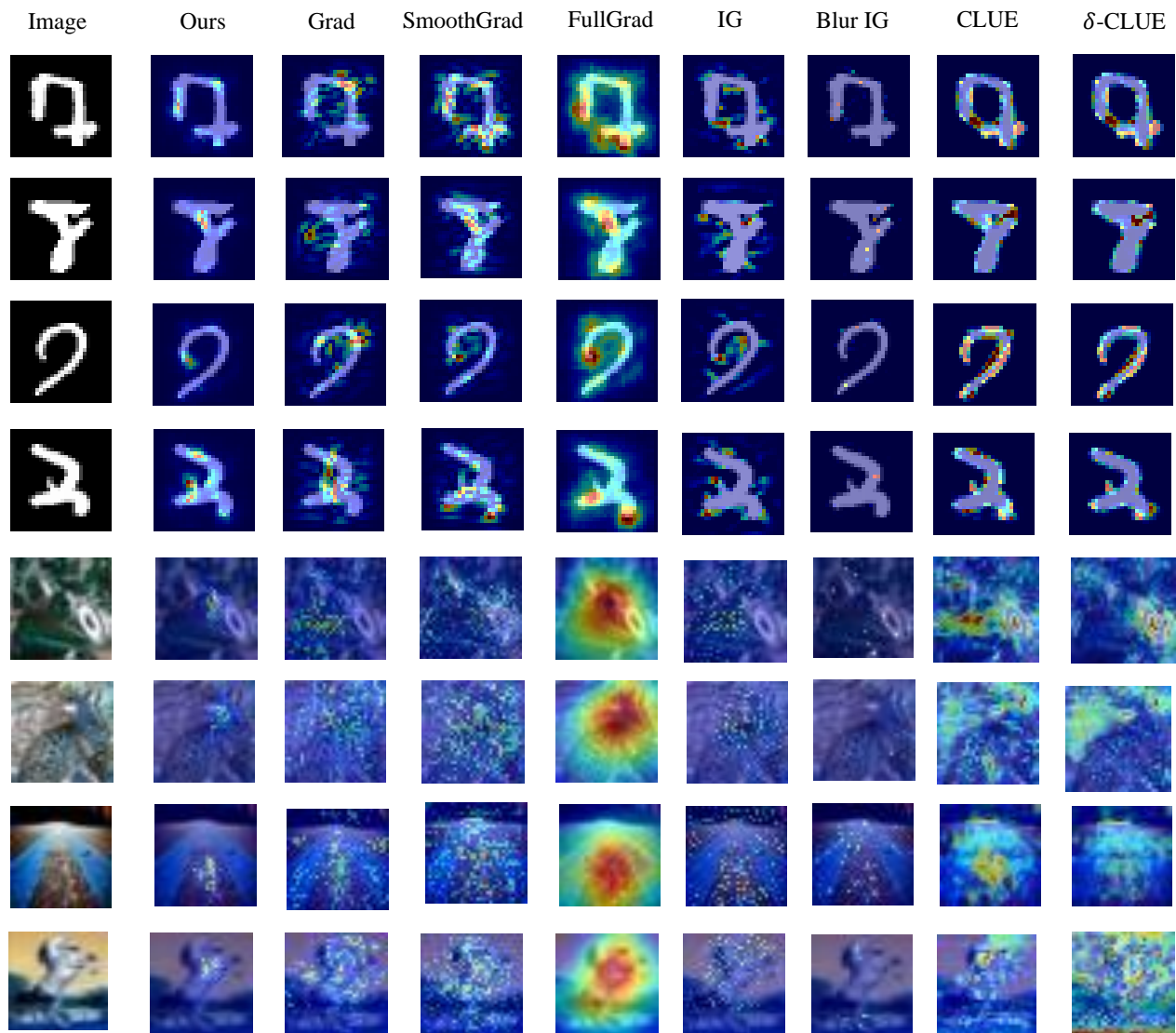
Figure 8. Additional examples of the uncertainty attribution maps for various methods across multiple datasets.