1. 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.

1.1. Essential Properties for Uncertainty Attribution

Adopted from the survey papers [1, 2, 6] 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.

1.2. 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. (1):

$$M_G(\boldsymbol{x}) = \left|\frac{\partial U}{\partial \boldsymbol{x}}\right|. \tag{1}$$

- **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. (2):

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

  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. (3):

$$M_{FG}(\boldsymbol{x}) = \psi\left(\left|\frac{\partial U}{\partial \boldsymbol{x}} \odot \boldsymbol{x}\right| + \sum_l \left|\frac{\partial U}{\partial \boldsymbol{b}_l} \odot \boldsymbol{b}_l\right|\right) \tag{3}$$

  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. (4):

$$M_{IG}(\boldsymbol{x}) = (\boldsymbol{x} - \boldsymbol{x}_0) \odot \int_0^1 \frac{\partial U(\boldsymbol{x}_0 + \alpha(\boldsymbol{x} - \boldsymbol{x}_0))}{\partial \boldsymbol{x}} d\alpha. \tag{4}$$

  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 [7] and Guided IG [4].

Based on the survey papers [1, 2, 6], we briefly summarize the properties satisfied by the aforementioned approaches in Table 1. In the next section, we will show the theoretical analysis of our proposed method.

Table 1. 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

1.3. Special Properties of UA-Backprop

Proposition 1.1. *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. (5):

$$M(\boldsymbol{x}) = \frac{1}{S}\sum_{s=1}^{S}\sum_{i=1}^{C} U_{z_i}^{s} M_i^{s}(\boldsymbol{x}) \tag{5}$$

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,

$$\begin{aligned}
&\sum_{(u,v)} M(\boldsymbol{x})[u,v] \\
&= \frac{1}{S}\sum_{s=1}^{S}\sum_{i=1}^{C} U_{z_i}^{s} \sum_{(u,v)} M_i^{s}(\boldsymbol{x})[u,v] \\
&= \frac{1}{S}\sum_{s=1}^{S}\sum_{i=1}^{C} U_{z_i}^{s} = \frac{1}{S}\sum_{s=1}^{S}\sum_{i=1}^{C}\sum_{j=1}^{C} c_{g_j^s \to z_i^s} U_{g_j} \\
&= \frac{1}{S}\sum_{s=1}^{S}\sum_{j=1}^{C}\left(\sum_{i=1}^{C} c_{g_j^s \to z_i^s}\right) U_{g_j} \\
&= \frac{1}{S}\sum_{s=1}^{S}\sum_{j=1}^{C} U_{g_j} = \frac{1}{S}\sum_{s=1}^{S} U = U
\end{aligned} \tag{6}$$

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.

2. 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.

2.1. Implementation Details and Training Hyperparameters

2.1.1 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 same architecture as [3]: 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.

2.1.2 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 $\boldsymbol{z} \to \boldsymbol{x}$, 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. (2), 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 $1e-3$.

2.2. Experiment Settings

2.2.1 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.

2.2.2 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 4.2.3. To evaluate model robustness, we retrain the model with the attention mechanism under limited data and

Table 2. 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
δ−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 4.2.

3. 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.

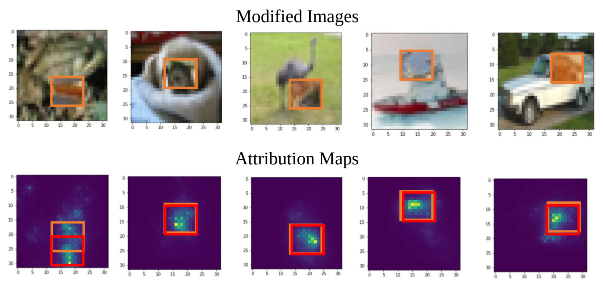

Figure 1. The anomaly detection examples. The red bounding boxes represent the predicted problematic regions while the orange bounding boxes are the ground truth.

As shown in Figure 1, 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 2 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.

4. Ablation Studies and Further Analysis

4.1. 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 3. 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 4, 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 4. 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

4.2. Different Experiment Settings for Uncertainty Attention Mitigation

4.2.1 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 3. 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.

4.2.2 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 5. 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.

4.2.3 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 6 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 5. 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 6. 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.

4.2.4 Input/Latent-space Attention for Uncertainty Mitigation

Table 7 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 2.2.2. Our results demonstrate that using UA maps as input-space attention yields similar performance compared to the results obtained through latent-space experiments.

4.3. 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 completeness 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 8, 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.

4.4. 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 7. 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 8. 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

$$M_i^s(\boldsymbol{x}) = \psi\left(\frac{\partial z_i^s}{\partial \boldsymbol{x}}\right) \tag{7}$$

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

$$M_i^s(\boldsymbol{x}) = \psi\left(\boldsymbol{x} \odot \frac{\partial z_i^s}{\partial \boldsymbol{x}}\right) \tag{8}$$

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,

$$\begin{aligned} &M_i^s(\boldsymbol{x}) \\ &= \psi\left((\boldsymbol{x} - \boldsymbol{x}_0) \odot \int_0^1 \frac{\partial z_i(\boldsymbol{x}_0 + \alpha(\boldsymbol{x} - \boldsymbol{x}_0), \boldsymbol{\theta}^s)}{\partial \boldsymbol{x}} d\alpha\right) \end{aligned} \tag{9}$$

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 9 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.

4.5. 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 10 are not good due to inadequate uncertainty quantification (UQ). By using more advanced single-network UQ methods, i.e. Laplacian approximation (LA) [5], 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.

4.6. 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 11 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 9. MURR and AUC-URR (AUC) of the blurring test for our proposed method with different approaches for $\boldsymbol{z} \rightarrow \boldsymbol{x}$. 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 10. 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 11. 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

4.7. 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.

5. Additional Examples

Figure 2 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.

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. 1

[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. 1

[3] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In *International Conference on Machine Learning*, pages 1183–1192. PMLR, 2017. 2

[4] 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. 1

[5] David J. C. MacKay. A Practical Bayesian Framework for Backpropagation Networks. *Neural Computation*, 4(3):448–472, 1992. 7

[6] 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. 1

[7] 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. 1

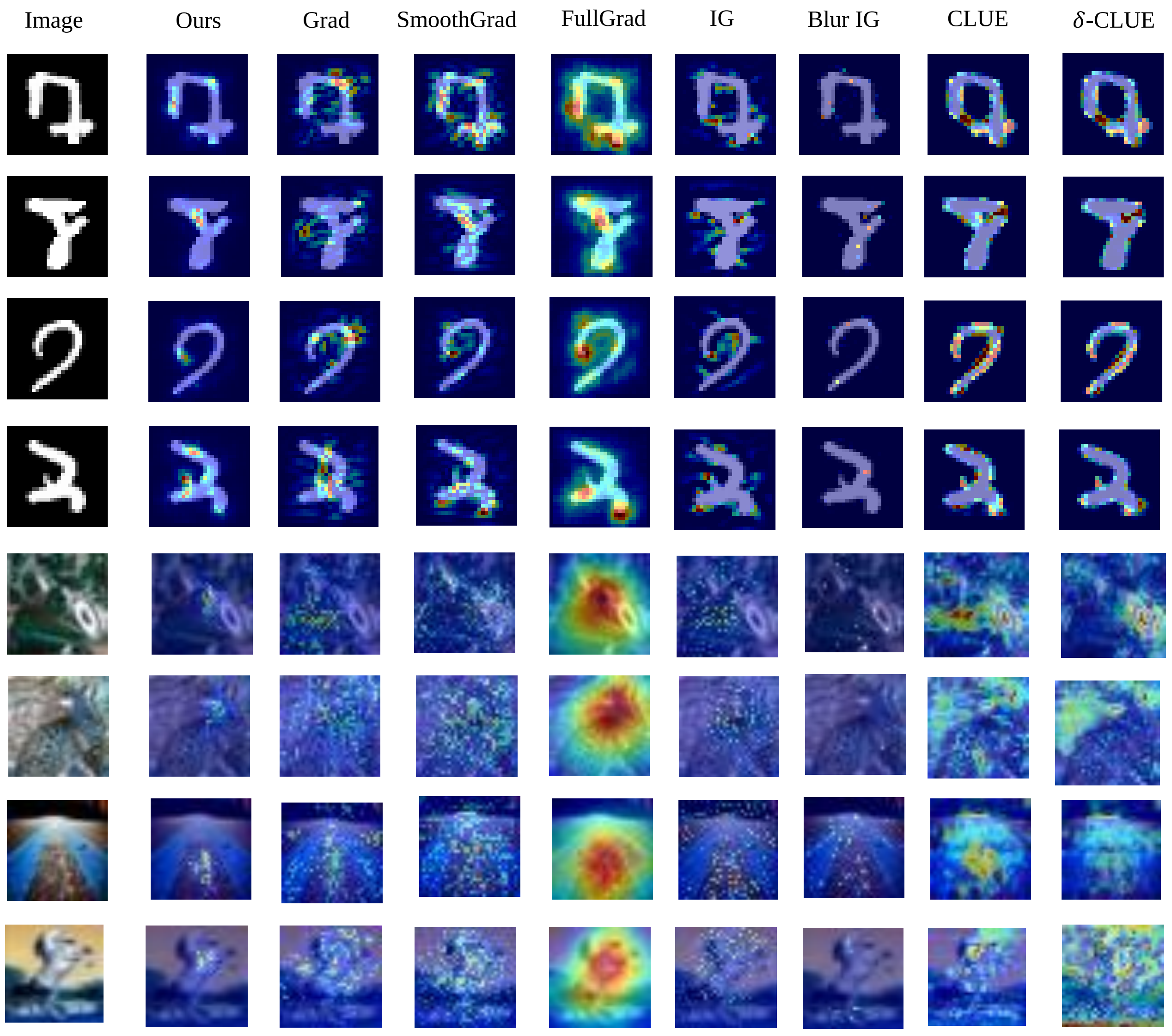

Figure 2. Additional examples of the uncertainty attribution maps for various methods across multiple datasets.