computers

MDPI

Article

Soft-Label Supervised Meta-Model with Adversarial Samples for Uncertainty Quantification

Kyle Lucke [1], Aleksandar Vakanski [1,2] and Min Xian [1,*]

[1] Department of Computer Science, University of Idaho, Idaho Falls, ID 83402, USA; klucke@uidaho.edu (K.L.); vakanski@uidaho.edu (A.V.)
[2] Department of Nuclear Engineering and Industrial Management, University of Idaho, Idaho Falls, ID 83402, USA
* Correspondence: mxian@uidaho.edu

Abstract: Despite the recent success of deep-learning models, traditional models are overconfident and poorly calibrated. This poses a serious problem when applied to high-stakes applications. To solve this issue, uncertainty quantification (UQ) models have been developed to allow the detection of misclassifications. Meta-model-based UQ methods are promising due to the lack of predictive model re-training and low resource requirement. However, there are still several issues present in the training process. (1) Most current meta-models are trained using hard labels that do not allow quantification of the uncertainty associated with a given data sample; and (2) in most cases, the base model has a high test accuracy. Therefore, the samples used to train the meta-model primarily consist of correctly classified samples. This leads the meta-model to learn a poor approximation of the true decision boundary. To address these problems, we propose a novel soft-label formulation that better differentiates between correct and incorrect classifications, thereby allowing the meta-model to distinguish between correct and incorrect classifications with high uncertainty (i.e., low confidence). In addition, a novel training framework using adversarial samples is proposed to explore the decision boundary of the base model and mitigate issues related to training datasets with label imbalance. To validate the effectiveness of our approach, we use two predictive models trained on SVHN and CIFAR10 and evaluate performance according to sensitivity, specificity, an F1-score-style metric, average precision, and the Area Under the Receiver Operating Characteristic curve. We find the soft-label approach can significantly increase the model's sensitivity and specificity, while the training with adversarial samples can noticeably improve the balance between sensitivity and specificity. We also compare our method against four state-of-the-art meta-model-based UQ methods, where we achieve significantly better performance than most models.

Keywords: uncertainty quantification; misclassification detection; meta-model; adversarial samples

check for updates

Academic Editors: Jeremy Straub, Hussain Mohammed Dipu Kabir, Syed Bahauddin Alam and Subrota Kumar Mondal

Received: 13 November 2024
Revised: 23 December 2024
Accepted: 24 December 2024
Published: 2 January 2025

Citation: Lucke, K.; Vakanski, A.; Xian, M. Soft-Label Supervised Meta-Model with Adversarial Samples for Uncertainty Quantification. *Computers* **2025**, *14*, 12. https://doi.org/10.3390/computers14010012

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

In recent years, deep learning-based models have been shown to outperform many classic machine learning algorithms. Moreover, in some areas, these models can perform better than humans [1–3]. However, traditional models are known to be overconfident and poorly calibrated [4]. This is a serious issue for high-stakes applications, e.g., healthcare, finance, nuclear energy, and autonomous driving. Hence, detecting mistakes is a crucial component for any model that will be used in such applications. To help mitigate this issue, a newly emerging sub-field, uncertainty quantification (UQ), has been developed.

UQ methods quantify the uncertainty associated with predictive models, thereby allowing the model to detect when its prediction is incorrect (misclassification detection) or when it encounters a sample that is not drawn from a distribution similar to the distribution of training samples (out-of-domain detection). When this happens, the model can take the appropriate action (e.g., transfer control of the system to a human user). For example, if a computer-aided diagnosis system detects the confidence of a prediction about a disease is low, it can alert a human medical doctor to this fact. This work is primarily concerned with misclassification detection.

In this work, we develop an independent neural network model (herein referred to as a meta-model) that is tasked with determining whether or not a classification model (herein referred to as a base model) has made a correct classification. Additionally, we propose a novel framework for training meta-model-based UQ methods that aims to address two shortcomings of the meta-model training process. (1) Ability to quantify uncertainty: Some previous works [5,6] use hard labels to train the meta-model. By doing this, the uncertainty of a given data sample cannot be adequately quantified and may be poorly calibrated. We propose a novel soft-label formulation for training meta-models to mitigate this issue. (2) Label imbalance in the meta-model training dataset: Well-trained base models that achieve high test accuracy (90%+) lead to an extremely imbalanced dataset for meta-model training. This causes the meta-model to learn a poor approximation of the true classification boundary, thereby causing the meta-model to have difficulty detecting misclassified samples. To address this problem, we propose a novel meta-model training framework that utilizes adversarial samples produced using a hard-label black-box adversarial attack to generate surrogate misclassified samples. Importantly, our framework is model-agnostic and can be applied to any meta-model.

To demonstrate the effectiveness of our framework, we use a meta-model method inspired by [5] and evaluate its ability to detect classification errors made by two base models (ResNet [7] and a small custom CNN) trained on different datasets. Our results show that the proposed soft-label formulation can drastically improve meta-model sensitivity and specificity, while the training framework can significantly improve the balance between sensitivity and specificity. Additionally, we compare our results against four state-of-the-art meta-model-based UQ methods where we significantly outperform most methods.

2. Related Work

In the following section, we present an overview of previously published work related to UQ, categorized according to their theoretical foundations. In addition, shortcomings in the context of misclassification detection and computational efficiency with respect to real-time and resource-constrained environments are briefly discussed.

2.1. Bayesian Methods

Bayesian methods approximate the posterior distribution of the weights of a neural network, thereby allowing the network to model uncertainty. In practice, a finite number of Monte Carlo (MC) samples are used to compute the mean and standard deviation of the predictions. Works such as Bayesian Neural Networks [8], MC-dropout [9] (MCD), and Deep Ensembles [10] (DEs) (which can be interpreted as Bayesian Model Averaging [11,12]) fall into this category.

While MCD is more computationally efficient to train than DEs, previous work [13] suggests the quality of uncertainty estimates obtained using MCD is poor due to the high correlation between the sampled dropout masks. To mitigate this issue, the authors of [13] propose using a fixed set of dropout masks whose overlap (and therefore correlation) can be controlled by a hyperparameter. While their method achieves performance comparable

to DEs at a small fraction of the computational training cost, sampling is still required to obtain the uncertainty estimate and the model size may be increased.

DEs and their variants [14,15] tend to outperform other Bayesian methods and are often considered a strong baseline for UQ. However, the quality of the uncertainty estimate depends on the diversity of models in the ensemble. Moreover, they require a large quantity of computational resources during training and inference, making them prohibitively expensive for applications that operate in real time or resource-constrained environments. There have also been works devised to train implicit ensembles that use weight sharing to increase the training and inference efficiency [16–18]. While these methods are more efficient, they typically still require multiple (albeit cheaper) forward passes. They also tend to perform slightly worse than an equivalent explicit ensemble because weight sharing can reduce the functional diversity of the ensemble members.

2.2. Density-Based Methods

Density-based methods measure the epistemic uncertainty of a new sample by using a kernel to compute the distance to the training data in feature space. Such methods typically suffer from feature collapse [19], a phenomenon where the feature extractor maps in-domain (ID) samples and out-of-domain (OOD) samples to the same area of feature space. Feature collapse typically occurs because such methods necessarily encourage the features of ID training samples to be mapped near a class centroid, so they have little control over samples that should be mapped far away from the training features. Hence, most methods in this field are concerned with regularizing the feature extractor to prevent feature collapse. Some works in this area [20–22] apply Spectral Normalization [23] to various layers of the feature extractor to regularize the feature space. Another work [19] proposes using a two-sided gradient penalty [24] originally proposed for Wasserstein GANs [25]. However, the authors of [22] argue Spectral Normalization is a faster and more effective regularization alternative that works well in practice. In either case, the regularization enforces a soft constraint on the Lipschitz constant of the model gradient thereby encouraging the feature extractor to be approximately distance-preserving, i.e., the distance in semantic space is approximately preserved in the learned feature representation.

Other methods [26] utilize contrastive learning (either self-supervised or supervised, depending on the availability of labels) to train the feature extractor on in-domain data. This implicitly encourages OOD data to be mapped far away from in-domain data. They then perform K-means clustering and independently model each cluster as a Gaussian. The Mahalanobis distance [27] between a new sample and the nearest cluster is used to compute the OOD score.

In either case, such methods are primarily concerned with OOD detection but could, in theory, be adapted for misclassification detection. However, because most misclassified samples are likely to be located near correctly classified samples, selecting an appropriate distance threshold for detection could be hard or may not be possible at all.

2.3. Meta-Model-Based Methods

Rather than modifying the original network to be able to provide uncertainty estimates, meta-model-based methods train a separate model dedicated to quantifying the uncertainty of the base model. Typically, these models use either the base model logits [6] or intermediate feature representations [5,28,29] as input data. Such methods offer two main advantages over Bayesian and density-based methods. (1) No re-training of the base model is required: This point is important in the context of already deployed applications where significant resources have already been dedicated to developing a base model that performs well. Moreover, re-training the base model may require substantial time and

resources. (2) Small resource footprint: Meta-models typically have fewer parameters than the base model. Thus, training them requires little time and resources and can be deployed alongside the base model with few additional resources. This point is important for models deployed in real time or resource-constrained environments (e.g., self-driving cars and portable medical devices).

In [5], intermediate feature representations are extracted from several layers of the base model and used to independently train linear classifier probes [30] to predict the ground-truth label associated with the input sample. The probe outputs are then concatenated together and serve as input to a Gradient Boosting Machine [31] (GBM) which is then trained to predict whether the label predicted by the base model is correct. In [29] the authors propose a Bayesian meta-model similar to [5]. However, several distinct differences exist: (1) the linear classifier probes are replaced with non-linear probes with several layers, (2) the GBM is replaced with a densely connected neural network, (3) the probes are trained in conjunction with the meta-model neural network, and (4) rather than directly predicting the misclassification made by the base model, the output of the meta-model is used to parameterize a Dirichlet distribution and is trained using the variational inference technique proposed in [32]. Several metrics derived from this distribution are then used as the misclassification detection score. In [28], a five-layer neural network named ConfidNet is trained using features extracted from the first layer of the classification head of the base model using their proposed loss function, True Class Probability (TCP). In [6], the authors propose Introspection-Net (I-Net), a three-layer neural network trained using the logits of the base model to perform misclassification detection. RED [33] trains a Gaussian process [34] (GP) model with a custom kernel to predict the residual between the misclassification label and the Maximum Class Probability [35] (MCP) of the base model predictions. The predicted residual is then used as the misclassification detection score.

2.4. Data-Imbalance

While much work has been conducted to help mitigate label imbalance in general machine learning and deep learning tasks [36,37], little work has been conducted in the context of meta-model-based UQ methods. In [38], the authors propose a meta-learning-based training framework where they carefully construct virtual training and testing sets with diverse levels of label imbalance and diverse input distributions, thereby encouraging the meta-model to generalize well to different label imbalance conditions and diverse input distributions. In [39], the authors propose a novel loss function, Steep Slope Loss, to help mitigate class imbalance by encouraging better separability of correct and incorrect classifications.

2.5. Dataset Augmentation and Sample Generation

In the context of OOD detection, many works have been proposed to generate or incorporate surrogate OOD samples for training more robust OOD detection models. Some methods [40–42] employ complex generative adversarial networks (GANs) to create samples that resemble OOD data, while others [43,44] devise strategies to incorporate a large auxiliary set of proxy OOD data into the detector training. GAN-based methods often have complex architectures, are computationally expensive to train, and still suffer from issues inherent to GANs (e.g., mode collapse, vanishing gradients [45], divergence during training). Incorporation-based strategies can suffer from contamination in the form of images that are semantically similar to the in-domain data [46] and may be biased towards the semantic information contained in the dataset, leading to difficulty detecting OOD data that are semantically different from the auxiliary data.

One recent method, VOS [47], takes a significantly different approach by fitting a class-conditional multivariate Gaussian distribution to the learned features of a classification model. Then, they sample so-called virtual outliers from the low-likelihood region of the feature space. However, some works suggest that the learned features may not follow a Gaussian distribution [48–50]. Moreover, their sampling procedure requires taking the inverse of the empirically estimated covariance matrix which may be numerically unstable if few samples are available [48] (a common scenario when training models using domain-specific datasets due to expensive labeling costs).

All of these methods were devised in the context of OOD detection and cannot be easily adapted for misclassification detection. They also do not take into consideration the boundary between correct and incorrect classifications.

2.6. Comparison with Previous Methods

I-Net [6] and ConfidNet [28] only use logits and features from the fully connected layer as inputs, respectively, and hence can effectively be considered BB models. While our results in this work indicate BB models typically outperform WB models, previous research [5] suggests WB models offer superior UQ performance when label noise is present. Label noise is somewhat prevalent in well-known benchmark natural image datasets [51] and very prevalent in medical image datasets, especially segmentation datasets [52]. Further, these models may miss important information contained in other layers. This is in contrast to our method which can be utilized as either a BB or WB method. ConfidNet uses TCP to prepare the labels for supervising its classifier which may have issues distinguishing between some correct and incorrect predictions (see Section 3.2 for a more in-depth discussion). Our proposed soft-label method mitigates this issue. The Linear Probe [5] (LP) and I-Net methods use hard labels for meta-model training, which does not allow the accurate quantification of the uncertainty associated with a given data sample (i.e., the confidence scores produced by the model may be poorly calibrated). In contrast to these methods, our model uses soft labels for training and thus can quantify the uncertainty associated with the base model prediction. Moreover, none of the aforementioned methods take measures to mitigate issues associated with label imbalance during training, while our method utilizes training with adversarial samples and thus does not suffer from these problems.

RED [33] uses a GP as the meta-model. Their results show that in addition to misclassification detection, the predicted variance of the GP can be used to detect OOD and adversarial samples. However, GPs are sensitive to hyperparameter initialization and may require many repeated trials to find good hyperparameters. RED is trained to predict errors between the MCP and a hard misclassification label. MCP does not adequately discriminate between correct and incorrect classification in instances where the base model prediction is "confidently wrong", e.g., when it predicts the wrong label with high probability. However, since the GP model has access to the softmax distribution predicted by the base model, this may mitigate some of these negative effects.

3. Materials and Methods

In the following section, we present an overview of the proposed model architecture (Section 3.1), discuss two major drawbacks with the current meta-model training process, and present our proposed solutions to these problems (Sections 3.2 and 3.3).

3.1. Overall Architecture

Existing meta-model-based UQ has achieved promising results; however, it suffers from two major drawbacks, biased training because of imbalanced training datasets and an insufficient ability to distinguish misclassifications and correct classifications with high

uncertainty. To address these issues, we propose a novel meta-model-based UQ method. A graphical overview of our proposed architecture is shown in Figure 1.

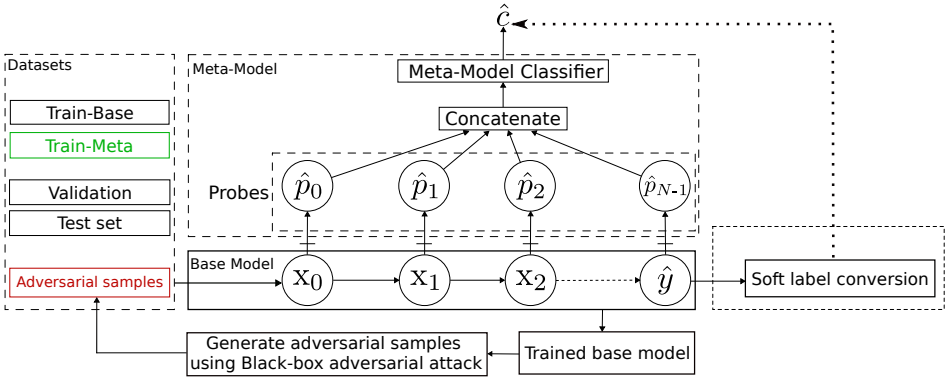


Figure 1. Overview of the proposed UQ architecture. A vertical line through the arrow body indicates gradients do not flow through that connection. x_i nodes represent intermediate feature values, $\hat{y}$ represents the probability distribution predicted by the base model, $\hat{p}_i$ nodes represent linear probes, and $\hat{c}$ is the soft label predicted by the meta-model.

The first major component in the architecture is the meta-model, which consists of N linear probes [30] and one meta-model classifier. The probes are independently trained to predict the ground-truth label of the data sample used to generate each feature map. The results of these probes are then concatenated together and used as input to the meta-model classifier. The meta-model is then trained to predict whether the base model correctly classified the provided data sample. In this work, we replaced the GBM used in [5] with a small four-layer feed-forward neural network that was trained to minimize the mean squared error between our proposed soft labels (presented in Section 3.2) and the confidence values predicted by the meta-model. The details of the meta-model architecture are outlined in Section 4.2.2.

This work evaluated two types of meta-models: black box (BB) and white box (WB). The BB meta-model only has access to the probe inserted into the final layer of the base model, while the WB meta-model has access to probes placed at various intermediate layers of the base model. If the user only has access to the logits or softmax values produced by the base model, the BB model can be used. On the other hand, if the user has full access to the base model, our WB approach offers great flexibility regarding where the probes are placed. In our experiments, the probes were placed after each max-pooling layer in the convolutional feature extractor (or residual block, in the case of ResNet [7]) and the flattened inputs to the base model classifier. Moreover, in either case, additional input features can easily be incorporated into the meta-model (e.g., in the case of a Bayesian-type base model, the estimated variance of the posterior distribution).

The second component is the generation of adversarial samples to mitigate label imbalance in the training set of the meta-model. A hard-label adversarial attack approach is applied to explore the decision boundary of the base model and generate data samples that are close to the boundary. This ensures the generated samples have a good portrayal of a base model's decision boundary and reduces the number of excessive and unnecessary training samples that would be generated using data augmentation approaches. Furthermore, it does not need any prior knowledge of the base model and could be used to efficiently generate adversaries for any deep learning-based base model. We generated adversarial samples using samples that were correctly classified by the base model to act as surrogate misclassified samples. The generated samples were then added to the meta-model training dataset.

The third component consists of the base model and the newly proposed soft-label conversion. The base model can connect to the meta-model using either the BB mode or WB mode. The soft-label conversion is proposed to create more informative soft labels that distinguish high-confidence correctly classified samples, low-confidence correctly classified samples, and misclassified samples.

Following the setup of [5], we divided the training data into three parts: Train-Base, Train-Meta, and Validation. Train-Base was used to train the base model, Train-Meta was used to train the meta-model (including the probes), and the Validation set was used for hyperparameter tuning and model selection during training. The motivation behind this setup was to separate the data used to train the base model and the meta-model. This ensured that severe overfitting of the base model to the training data did not affect meta-model performance. For example, in the most extreme case, the base model may achieve 100% training accuracy but poor test accuracy. Hence, the training dataset provides a bad estimate of the true test accuracy and misleads the training of the meta-model.

3.2. Soft-Label Conversion

Using hard labels to train the meta-model only allows the meta-model to perform misclassification detection and does not allow the quantification of the uncertainty associated with the prediction of the base model. Moreover, it has been shown that soft labels are more informative than hard labels and are commonly used in knowledge distillation [53]. Motivated by these insights, we propose using soft labels to supervise the meta-model during training.

One way to generate the soft label is to use the normalized probability predicted by the base model that a data sample belongs to the ground-truth class. This corresponds to using TCP [28] to supervise the meta-model. However, this formulation cannot adequately distinguish between correct classifications with high uncertainty and incorrect classifications. For example, consider a three-class classification problem. Without loss of generality, let the ground-truth label of two different data samples be class 0. Suppose the probability distributions predicted by the base model are $[0.4, 0.3, 0.3]$ and $[0.4, 0.6, 0.0]$, respectively. TCP would use 0.4 as the supervision label for both data samples. However, the first data sample is correctly classified while the second is not. To address this issue, we propose a new soft-label generation approach defined by

$$c = \begin{cases} P(y|x), & \text{if } \hat{y} = y \\ -(1 - P(y|x)), & \text{if } \hat{y} \neq y \end{cases}, \tag{1}$$

where c is the generated soft label, y is the ground truth, $\hat{y}$ is the predicted hard label of the base model, x is the input, and $P(y|x)$ is the predicted probability (initial soft label) of the base model that the data sample x belongs to the ground-truth class, y. Put simply, if the prediction is correct ($\hat{y} = y$), $P(y|x)$ is used as the final soft label; if it is an incorrect prediction, the soft label is mapped to a negative value. The separate equations distinguish correct and incorrect classifications and allow the meta-model to differentiate between highly confident correct classifications (c near 1), low-confidence correct classifications (c close to but greater than 0), low-confidence incorrect classifications (c close to but less than 0), and high-confidence incorrect classifications (c near -1).

3.3. Meta-Model Training with Adversaries

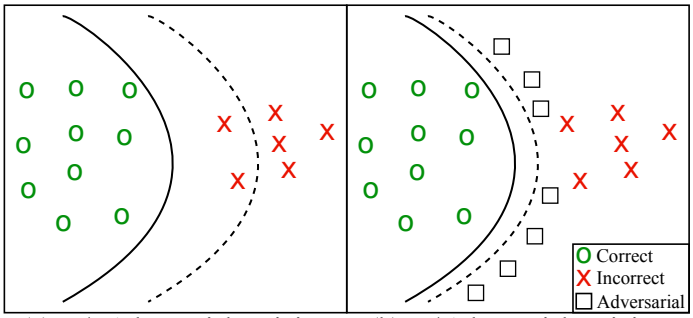
Training a meta-model to perform misclassification detection of a well-trained base model suffers from extreme data imbalance because most data samples will be correctly classified. This imbalance could cause the meta-model to learn a poor estimate of the true

decision boundary of the base model. Hence, the meta-model will have difficulty detecting misclassified data samples. To help mitigate this issue, we propose a novel meta-model training framework inspired by adversarial training [54]. We generate adversarial samples for correctly classified data samples by exploring the decision boundary of the base model. The adversarial samples act as surrogates for misclassified samples during the meta-model training. Specifically, the generation of adversarial samples is formulated as an untargeted hard-label black-box attack of a base model,

$$x^* = \arg\min_x \ \mathrm{D}(x, x_0)_\infty \text{ such that } f(x) \neq y \tag{2}$$

where $\mathrm{D}(x, x_0)_\infty$ is the distance between a clean correctly classified sample x_0 and its adversary x according to the L_∞ norm, $f(x)$ denotes a base model, and y is the true class label of x_0. A two-stage optimization [55,56] is applied to find adversarial samples within a user-supplied distance ϵ from the decision boundary of the correct class. The optimization is posed as a discrete problem, where search directions are restricted to the vertices of the L_∞ norm ball around the clean sample. The first stage performs a fast check step to eliminate search directions that are suboptimal (i.e., that do not result in an adversarial sample). The second stage then starts from a known good search direction that results in an adversarial sample and performs a binary search in that direction to find a sample closer to the decision boundary. Hence, using a small distance allows us to generate misclassified samples near the decision boundary. Each correctly classified sample in Train-Meta was used to generate an adversarial negative sample. We also included all clean misclassified samples in the Train-Meta partition of the dataset. By incorporating an adequate number of adversarially generated misclassified samples that were sufficiently close to the decision boundary, the meta-model could learn a significantly better approximation of the true decision boundary.

Figure 2 graphically depicts how the proposed training with adversarial samples framework can improve the decision boundary learned by the meta-model. Due to their proximity to the decision boundary, the adversarial samples are also likely to lie in a low-density region. Hence, our training approach encourages the meta-model to learn a much smoother decision boundary, thereby leading to a more robust model. While the generated adversarial samples may not resemble legitimate reasons for misclassification, our empirical results indicated that the proposed training mechanism could result in better meta-model performance.



(a) W/o Adversarial Training (b) W/ Adversarial Training

Figure 2. Graphical depiction of how the proposed training framework improves the learned decision boundary of a meta-model. (**a**) Without using adversarial samples during training, the decision boundary learned by the meta-model (dashed line) is a poor approximation of the true decision boundary between correctly and incorrectly classified data samples (solid line) due to the imbalance between the number of correctly and incorrectly classified samples. (**b**) Adding adversarial samples near the decision boundary as surrogate misclassified samples causes the decision boundary learned by the meta-model to be a better approximation of the true decision boundary.

4. Results

In this section, we provide the details for our experimental setup, verify the effectiveness of each proposed component, and compare the performance of our model against four previously proposed state-of-the-art methods. Specifically, in Section 4.1 we present the datasets, models, and metrics used to evaluate our method, Section 4.2 details model training, hardware, and libraries, Sections 4.3 and 4.4 present results related to the performance of our improvements, and Section 4.5 compares our method with others.

4.1. Datasets, Base Models, and Metrics

4.1.1. Datasets

We evaluated our proposed method using the SVHN [57] and CIFAR10 [58] datasets with the standard training and test splits. Note that for the SVHN dataset, we did not utilize the extra training set, only the standard training split. Hence, there were 73,257 training samples and 26,032 test samples for the SVHN dataset and 50,000 training samples and 10,000 test samples for the CIFAR10 dataset. The training data were then divided into Train-Base, Train-Meta, and Validation partitions. Table 1 shows the number of samples and the percentage of the total number of training samples in each partition for the respective datasets. The proportion of the Train-Meta partition was selected to contain approximately 10,000 data samples.

Table 1. The number of samples and percentage of the total training data for the respective partitions.

Dataset	Train-Base	Train-Meta	Validation
SVHN	55,382 (75.60%)	10,550 (14.40%)	7325 (10.00%)
CIFAR10	34,650 (69.30%)	10,350 (20.70%)	5000 (10.00%)

4.1.2. Base Models

The base model used for the CIFAR10 dataset was ResNet-32 V1 [7]. For the SVHN dataset, we used the small custom CNN architecture described in [28]. The motivation behind using these base models was their ability to achieve at least 90% test accuracy. To ensure fairness, each meta-model used the same trained base model.

4.1.3. Metrics

To evaluate performance, we used three threshold-agnostic and three threshold-specific metrics. The threshold agnostic metrics used were the Area Under the Receiver Operating Characteristic curve (AUROC) and average precision (AP). The AUROC can be interpreted as the probability that a model will rank a positive sample above a negative sample, given a randomly chosen positive and negative sample. AP is a summary measure of the precision–recall curve. It is computed as the weighted mean of the precision where the weights represent the difference between the recall at the current and previous threshold. To assess the meta-model's ability to detect correct classifications and misclassifications, we evaluated model performance using both correct and incorrect classifications as the positive class, denoted with "-S" and "-E", respectively (e.g., AP-S denotes the average precision with correct classifications as the positive class, while AP-E denotes average precision with misclassification as the positive class). The threshold-specific metrics used were sensitivity (Sens), specificity (Spec), and a summary metric similar to the F_β metric [59], which we denote F-SS_β:

$$\text{F-SS}_\beta = (1+\beta^2)\frac{\text{sensitivity} \cdot \text{specificity}}{(\beta^2 \cdot \text{sensitivity}) + \text{specificity}} \tag{3}$$

where the parameter β is a non-negative real number. β can be adjusted to give more weight to specificity ($\beta > 1$) or sensitivity ($\beta < 1$). Throughout the rest of this work, we used $\beta = 1$.

The motivation behind using sensitivity and specificity was to measure the percentage of true positives (correct classifications) and true negatives (incorrect classifications) detected by the meta-model, respectively. The definitions of the rest of the metrics are summarized in Table 2. In the comparison against other state-of-the-art models, we additionally report the average inference time per sample to evaluate model efficiency.

Table 2. Quantitative metrics for misclassification detection. TP: true positives, FP: False positives, TN: true negatives, FN: false negatives, R_n and P_n indicate recall and precision at the nth threshold, respectively.

Metric	Definition
Sensitivity	$\frac{\text{TP}}{\text{TP+FN}}$
Specificity	$\frac{\text{TN}}{\text{TN+FP}}$
AP	$\sum_n (R_n - R_{n-1}) P_n$

4.1.4. Threshold Selection

Threshold selection for the relevant metrics was achieved by evaluating model performance across several thresholds on the Validation dataset. Specifically, we generated thresholds at a 0.01 interval between the minimum and maximum value of the predictions and selected the threshold that maximized the F-SS_1 metric. For models trained with hard labels and a sigmoid activation, we interpreted the model output as a probability value and used a threshold of 0.5.

4.2. Experimental Setup

4.2.1. Base Model Training

The CNN used for the SVHN dataset was trained for 100 epochs with a batch size of 128 using the SGD optimizer [60] with a learning rate of 1×10^{-3}, a momentum of 0.9, and a weight decay of 1×10^{-4}. The input data were scaled to be in $[-1, 1]$. No data augmentation was used.

ResNet-32 was trained on CIFAR10 for 200 epochs with a batch size of 128 using the SGD optimizer with a learning rate of 0.1, a momentum of 0.9, and a weight decay of 1×10^{-4}. Additionally, we used a multi-step learning rate scheduler with a gamma of 0.1 to reduce the learning rate at epochs 100 and 150. The input data were first scaled to be in $[0, 1]$ and then normalized using a mean of $[0.485, 0.456, 0.406]$ and a standard deviation of $[0.229, 0.224, 0.225]$. The data augmentation transforms used were random horizontal flips with a probability of 0.5, and random cropping of size 32×32 with four pixels of constant padding using a fill value of zero. Both models were trained using the cross-entropy loss [61]. Under these setups, we achieved a test accuracy of 95.44% and 90.82% on the SVHN and CIFAR10 datasets, respectively.

4.2.2. Meta-Model Architecture and Training

For the proposed method, each probe consisted of a single densely connected linear layer with a softmax activation function. The input size of the probes varied depending on the input feature map size, and the output size was equal to the number of semantic classes in the dataset. The probes were trained for 200 epochs with a batch size of 128 using the Adam [62] optimizer with a learning rate of 1×10^{-4} according to the cross-entropy loss. During training, the best model according to the validation accuracy was retained.

The meta-model classifier was a four-layer neural network with dropout [63] after each layer and ReLU [64] activation functions. See Table 3 for specific layer sizes. The meta-model input size varied depending on the number of probes used and the number of ground-truth semantic classes in the dataset. The final layer of the classifier trained using hard and soft labels utilized sigmoid and hyperbolic tangent activation functions, respectively. The network was trained for 200 epochs using the Adam [62] optimizer with a learning rate of 1×10^{-3} and a batch size of 128. During training, the best model according to validation loss was retained. The best dropout rate according to the validation loss was found by searching over dropout rates $\in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8\}$. We used the binary cross-entropy [61] and mean squared error loss for the meta-models trained with hard and soft labels, respectively. The image data were preprocessed in the same way as during the base model training. No data augmentation was used for either dataset during meta-model training. Additionally, the probe outputs were normalized $\in [0, 1]$ using the softmax function before being used as input to the meta-model classifier.

The runtime of the WB method was linear with respect to the number of probes used, and the overall runtime of the probes would likely be dominated by the probe nearest to the input layer as they tended to have the largest feature map size and thus would have the largest probe input size. The input size of the meta-model classifier would also increase slightly with a larger number of probes and may also need slightly larger intermediate layers to achieve good performance for the increased input size, but this would be a negligible increase in runtime with respect to the probes.

Table 3. Meta-model architectures. BB: black box, WB: white box, N_{probes} denotes the number of probes used for the respective base model, N_{classes} denotes the number of classes in the dataset used to train the base model (e.g., 10 for CIFAR10).

Model	Layers
BB	N_{classes} - 150 - 100 - 20 - 1
WB	$N_{\text{probes}} \cdot N_{\text{classes}}$ - 300 - 200 - 40 - 1

4.2.3. SOTA Training

All state-of-the-art models were trained using the Meta-Train partition. Where possible, we adopted the same training setups (e.g., optimizer, learning rate, probe placement, loss function, etc.) as described in the original papers. For all methods except LP [5] and I-Net, we used the publicly available source code that accompanied the paper. We were unable to obtain the implementation of LP used in its authors' paper, so we used our implementation. For I-Net, we used the implementation provided in the RED [33] repository. For the Dirichlet model [29], we used the Max Probability score described in that paper as the authors' results suggested it performed best for misclassification detection. Each model was trained using the same batch size as our models (128). We did not apply the proposed training framework to any models besides ours. The original training setups of the other methods did not employ any techniques to mitigate class imbalance.

4.2.4. Adversarial Training Settings

The surrogate negative samples utilized in the training with adversaries were generated using the untargeted version of RayS [55] with $\epsilon = 0.03$ and a query limit of 20,000. The RayS attack achieved a success rate of 96.75% and 100% for the SVHN and CIFAR10 datasets, respectively.

4.2.5. Statistical Validation of Results

To validate the statistical significance of the of our proposed improvements and performance against state-of-the-art models, we used McNemar's test [65] with continuity correction [66].

4.2.6. Libraries and Hardware

Experiments were conducted using PyTorch [67] version 1.10.0, and CUDA version 11.8 on a single NVIDIA RTX 6000 GPU with 49 GB of memory utilizing an AMD Epyc 7763 CPU with 1 TB of RAM. McNemar's Test was conducted using version 0.23.3 of the MLxtend [68] library.

4.3. Effectiveness of Soft Labels

In this section, we validate the effectiveness of our proposed soft-label formulation. The results on the SVHN and CIFAR10 datasets are shown in Table 4. For the BB model on the SVHN dataset, the sensitivity and AUC decreased by 0.12 and 0.01, respectively, while the specificity, F-SS_1, and AP-E increased by 0.63, 0.49, and 0.01, respectively, while AP-S remained competitive. For the WB meta-model, we saw a decrease of 0.15 in sensitivity and an increase of 0.84, 0.84, 0.04, and 0.01 in specificity, F-SS_1, AP-E, and AUC, respectively, while AP-S remained competitive.

Table 4. Effectiveness of the proposed soft-label formulation for the SVHN and CIFAR10 datasets. HL: hard label, SL: soft label. The best values are in bold. * indicates the difference in results is statistically significant according to McNemar's test, with a p-value < 0.05.

SVHN							
Type	**Method**	**Sens**	**Spec**	**F-SS_1**	**AP-S**	**AP-E**	**AUROC**
BB	HL	**0.989**	0.236	0.381	**0.995**	0.411	**0.925**
	SL *	0.868	**0.869**	**0.868**	0.994	**0.421**	0.915
WB	HL	**1.000**	0.000	0.000	**0.993**	0.303	0.890
	SL *	0.846	**0.839**	**0.843**	**0.993**	**0.339**	**0.904**
CIFAR10							
Type	**Method**	**Sens**	**Spec**	**F-SS_1**	**AP-S**	**AP-E**	**AUROC**
BB	HL	**0.985**	0.188	0.316	**0.989**	0.470	**0.908**
	SL *	0.837	**0.843**	**0.840**	0.988	**0.472**	0.904
WB	HL	**0.981**	0.232	0.375	**0.985**	0.433	0.881
	SL *	0.816	**0.820**	**0.818**	**0.985**	**0.435**	**0.883**

Looking at the BB meta-model for the CIFAR10 dataset, we saw a decrease of 0.15 in sensitivity and an increase of 0.66 and 0.52 in specificity and F-SS_1, respectively, while all other metrics remained competitive. For the WB meta-model, we saw a decrease of 0.17 in sensitivity and an increase of 0.59 and 0.44 in specificity and F-SS_1, respectively, while all other metrics remained competitive.

Overall, using soft labels to train the meta-model resulted in significantly improved performance for all of the threshold-based metrics and most of the threshold-agnostic metrics. While in some cases the sensitivity was reduced considerably, there was a noticeably better balance between sensitivity and specificity (as evidenced by the improvement in the F-SS_1 metric). Moreover, we found that there was a statistical difference in the results, as indicated by McNemar's test with a p-value < 0.05.

4.4. Effectiveness of Meta-Model Training with Adversaries

In Table 5, we evaluated how long it took to generate the adversarial samples used in the training with adversaries. Interestingly, the total generation time for the SVHN dataset was significantly lower than for the CIFAR10 dataset, despite the fact that the CIFAR10 model was much larger. This was due to the larger average number of queries necessary to generate the adversarial samples. While the generation time was large, the adversarial samples only needed to be generated once before training. RayS is a hard-label black-box attack; if the user has full access to the base model, more efficient white-box attack methods could be used. It is also worth noting that more efficient implementations of RayS exist [56]. Moreover, the publicly available implementation of RayS only allows early stopping in the single-sample implementation, so we were only able to generate a single adversarial sample at a time. Hence, an implementation that allows for early stopping when operating on a batch of samples would greatly reduce the generation time. Early stopping terminates the adversarial attack once a sample within ϵ is found, otherwise, the algorithm continues searching for a closer sample.

The results in Table 6 show that using the proposed training framework on the BB model with hard labels resulted in a 0.04 and 0.07 decrease in sensitivity and AP-E, respectively, and a 0.41 and 0.39 increase in specificity and F-SS_1, respectively, while all other metrics remained competitive. Using the proposed training framework with soft labels resulted in a 0.04 and 0.02 decrease in sensitivity and AP-E, respectively, and a 0.04 increase in specificity, while all other metrics remained competitive.

The results in Table 6 show that using the proposed training framework on the WB model with hard labels resulted in a 0.08 and 0.02 decrease in sensitivity and AP-E, respectively, and a 0.72 and 0.81 increase in specificity and F-SS_1, respectively. Using the proposed training framework with soft labels on the WB resulted in a 0.04 decrease in AP-E, and a 0.03 and 0.01 increase in specificity and AUC, respectively, while all other metrics remained competitive.

The results in Table 7 show that using the proposed training framework on the BB model with hard labels resulted in a 0.03 decrease in sensitivity and AP-E, and a 0.29 and 0.32 increase in specificity and F-SS_1, respectively, while all other metrics remained competitive. Using the proposed training framework with soft labels resulted in a 0.03 decrease in AP-E, while all other metrics remained competitive.

The results in Table 7 show that using the proposed training framework on the WB model with hard labels resulted in a 0.09 and 0.04 decrease in sensitivity and AP-E, respectively, and a 0.49 and 0.42 increase in specificity and F-SS_1, respectively, while all other metrics remained competitive. Using the proposed training framework on the WB model with soft labels resulted in a 0.02 decrease in AP-E and a 0.01 increase in sensitivity, specificity, and F-SS_1, while all other metrics remained competitive.

Table 5. Time taken to generate the adversarial samples used in the training with adversaries. Total Time indicates the time, in hours, taken to generate all samples. Generation Time denotes the average and standard deviation of the time taken to generate a single adversarial sample. Note that this indicates the time taken for all generation attempts, including samples for which RayS was unable to generate an adversarial sample within the given ϵ.

Dataset	Total Time (h)	Generation Time (s)
SVHN	11.700	4.213 ± 7.372
CIFAR10	8.915	3.397 ± 4.184

Table 6. Effectiveness of training with adversaries for the SVHN dataset. TA: training with adversaries. The best values are in bold. * indicates the difference in results is statistically significant according to McNemar's test, with a p-value < 0.05.

Type	Method	Sens	Spec	F-SS$_1$	AP-S	AP-E	AUROC
BB	HL w/o TA	**0.989**	0.236	0.381	**0.995**	**0.411**	**0.925**
	HL w/ TA *	0.945	**0.647**	**0.768**	**0.995**	0.340	0.917
	SL w/o TA	**0.868**	0.869	**0.868**	**0.994**	**0.421**	**0.915**
	SL w/ TA *	0.826	**0.906**	0.864	**0.994**	0.399	**0.915**
WB	HL w/o TA	**1.000**	0.000	0.000	**0.993**	**0.303**	0.890
	HL w/ TA *	0.919	**0.720**	**0.807**	0.992	0.282	**0.893**
	SL w/o TA	**0.846**	0.839	0.843	**0.993**	**0.339**	**0.904**
	SL w/ TA *	0.837	**0.867**	**0.852**	0.992	0.304	0.894

Table 7. Effectiveness of training with adversaries for the CIFAR10 dataset. The best values are in bold. * indicates the difference in results is statistically significant according to McNemar's test, with a p-value < 0.05.

Type	Method	Sens	Spec	F-SS$_1$	AP-S	AP-E	AUROC
BB	HL w/o TA	**0.985**	0.188	0.316	**0.989**	**0.470**	**0.908**
	HL w/ TA	0.952	**0.477**	**0.636**	**0.989**	0.438	0.906
	SL w/o TA	0.837	**0.843**	0.840	**0.988**	**0.472**	**0.904**
	SL w/ TA	**0.840**	**0.843**	**0.842**	0.987	0.447	0.899
WB	HL w/o TA	**0.981**	0.232	0.375	**0.985**	**0.433**	0.881
	HL w/ TA *	0.891	**0.723**	**0.798**	**0.985**	0.393	**0.886**
	SL w/o TA	0.816	0.820	0.818	**0.985**	**0.435**	**0.883**
	SL w/ TA *	**0.829**	**0.833**	**0.831**	0.983	0.419	0.880

Overall, the proposed training framework could significantly improve sensitivity and specificity when the metrics were unbalanced (e.g., in the hard-label case). However, when the metrics were already balanced (e.g., in the soft-label case), the performance remained nearly the same. We additionally found that the difference in results was statistically significant, except for the BB model on the CIFAR10 dataset.

4.5. Comparison with State of the Art

In this section, we compare our method against four state-of-the-art meta-model-based uncertainty quantification methods, I-Net [6], Dirichlet [29], LP [5], and ConfidNet [28]. We do not report results for RED [33] because its proposed evaluation framework is much different than ours. The results for the SVHN and CIFAR10 datasets are shown in Tables 8 and 9, respectively.

Table 8. State-of-the-art results for the SVHN dataset. The best values are in bold. * indicates the difference in results is statistically significant with respect to all other state-of-the-art models according to McNemar's test, with a p-value < 0.05. Values in the **Time** column are the average inference time per sample in seconds.

Method	Sens	Spec	F-SS$_1$	AP-S	AP-E	AUROC	Time (s)
I-Net [6]	0.753	0.891	0.816	0.993	0.311	0.898	2.26×10^{-6}
Dirichlet [29]	0.771	0.878	0.821	0.992	0.269	0.885	4.834×10^{-7}
LP [5] (BB)	1.000	0.000	0.000	0.990	0.176	0.853	1.950×10^{-2}
LP [5] (WB)	0.997	0.035	0.067	0.990	0.196	0.846	1.227×10^{-2}
ConfidNet [28]	0.814	**0.909**	0.859	**0.995**	**0.413**	**0.925**	$\mathbf{1.342 \times 10^{-8}}$
Novel UQ (ours, WB) *	**0.837**	0.867	0.852	0.992	0.304	0.894	8.266×10^{-4}
Novel UQ (ours, BB) *	0.826	0.906	**0.864**	0.994	0.399	0.915	6.566×10^{-4}

Table 9. State-of-the-art results for the CIFAR10 dataset. The best values are in bold. * indicates the difference in results is statistically significant with respect to all other state-of-the-art models according to McNemar's test, with a p-value < 0.05.

Method	Sens	Spec	F-SS$_1$	AP-S	AP-E	AUROC	Time (s)
I-Net [6]	0.796	0.863	0.828	0.988	0.422	0.897	5.548×10^{-6}
Dirichlet [29]	0.792	0.856	0.823	0.988	0.427	0.897	2.945×10^{-7}
LP [5] (BB)	1.000	0.000	0.000	0.981	0.304	0.845	4.286×10^{-3}
LP [5] (WB)	0.977	0.123	0.219	0.982	0.299	0.846	5.113×10^{-3}
ConfidNet [28]	0.788	**0.881**	0.832	**0.990**	**0.449**	**0.905**	$\mathbf{3.675 \times 10^{-8}}$
Novel UQ (ours, WB) *	0.829	0.833	0.831	0.983	0.419	0.880	2.347×10^{-4}
Novel UQ (ours, BB) *	**0.840**	0.843	**0.842**	0.987	0.447	0.899	5.999×10^{-4}

On the SVHN dataset (Table 8), our BB model achieved competitive or superior performance over most methods in terms of all metrics, except ConfidNet [28], where our method was 0.01 lower in terms of AP-E and AUC. On the CIFAR10 dataset (Table 9), our BB model achieved significantly better performance than most other methods except ConfidNet, where it achieved worse performance in terms of specificity (-0.04); however, it achieved a higher sensitivity (+0.05) and a better balance between sensitivity and specificity. Additionally, we found that the difference in results for both our BB and WB model was statistically significant with a p-value < 0.05. We also investigated the average inference time (in seconds) for each method. While we found that both WB and BB methods performed several orders of magnitude slower than most methods, except the LP methods, the inference time of any reasonably sized base model would dominate the overall inference runtime. Hence, the meta-model inference runtime would be relatively negligible.

5. Discussion

The results show that the proposed soft-label formulation (Table 4) can significantly improve performance over hard labels, while the results of training with adversaries (Tables 6 and 7) indicate that the proposed training framework can significantly improve the sensitivity and specificity in cases where the results are unbalanced (e.g., the hard label models). However, little improvement is shown in cases where the sensitivity and specificity are already balanced.

In most of our experiments, we achieved significantly better specificity at the cost of some sensitivity. For most high-stakes applications, this is desirable as false positives (incorrectly classified samples predicted to be correctly classified) are to be avoided at all costs. In the context of self-driving vehicles, such mistakes could result in significant damage to property, pedestrians, or vehicle passengers. On the other hand, when applied to healthcare applications, false positives (e.g., incorrectly reporting the absence of a disease) could result in delayed treatment or loss of life. In practice, the threshold could be tuned for the desired application using the F-SS$_\beta$ metric with a proper value of β to achieve the desired trade-off. The false positive rate at a desired true positive rate could also be used.

There are several shortcomings in this work. We present each shortcoming in its own subsection for better readability.

5.1. Sensitivity Analysis of ϵ

In this work, the ϵ parameter for RayS was chosen empirically using a rough search. A more in-depth investigation of parameter sensitivity for the training framework should be conducted in the future. Utilizing a mixture of adversaries generated at different distances during training, reminiscent of the multi-adversary robust self-training proposed in [69], may be able to produce a smoother decision boundary for the meta-model classifier and provide even greater performance. It may also be worth investigating how different adversarial attacks affect meta-model performance.

5.2. Soft-Label Improvements

In our preliminary investigation, we found that the soft-label distribution of the correct and incorrect samples were heavily skewed towards 1 and -1, respectively. This means that the meta-model may have difficulty detecting correctly classified samples with low confidence values and vice versa. One way to mitigate this problem could be to craft special perturbations applied to the input samples that decrease or increase the probability predicted by the base model for correctly and incorrectly classified samples, respectively, in a manner reminiscent of adversarial attacks.

5.3. Application to Other Datasets, Models, and Tasks

Further investigation of model performance when applied to larger (like ImageNet [70]) or more fine-grained datasets (like iNaturalist [71]), different base models, different tasks (e.g., segmentation, object detection, regression), and other image modalities (e.g., medical images, LiDAR) is necessary. While we developed this method for misclassification detection in the context of image classification, it is reasonable to assume our method could be easily adapted for segmentation by replacing the linear layers in the probes and classification model with convolutional operations.

The proposed method could also reasonably be used for regression by changing the training scheme of the probes and meta-model appropriately. Namely, the probes could be trained to predict the regression value associated with the input sample and the meta-model could be trained to predict the error between the base-model prediction and the target values. Changes would also need to be made to the adversarial sample

generation. Adversarial attacks in the context of regression models are an underexplored area of research; however, methods do exist [72,73]. Since a well-trained regression model would produce low residuals (i.e., the difference between the prediction and true regression values will be small) it is likely the meta-model would underestimate the error of samples with large deviation from the target values, as few of them would exist in the meta-model training data. Thus, the adversarial sample generation would involve intentionally creating samples that have a large residual and using them during training.

Different imaging modalities, like medical images, may present additional challenges due to the presence of noise and large variations in the object of interest in terms of anatomy, size, and shape. Moreover, we only tested the performance of this method on misclassification detection. How to apply our method to out-of-domain detection is worth investigating in the future as well.

5.4. Better Control over Generated Adversarial Samples

When applying the proposed training framework to larger or more fine-grained datasets, the decision boundary of the base model will be complex. Generating the adversarial samples in an untargeted manner may bias the meta-model towards certain classes. For example, it may be the case that the closest decision boundary for many correctly classified samples from class 0 is near class 1. In this case, most of the generated adversarial samples will likely be incorrectly classified by the base model as class 1. Hence, the meta-model may struggle to identify misclassified samples that belong to other classes. An investigation of misclassification detection at the class level would also be an interesting future research direction.

5.5. Separate Partition for the Meta-Model

Our current approach assumes access to an additional set of $\approx 10,000$ data samples not used during base model training. In specialized domains where obtaining data labels is expensive (e.g., biomedical image processing), it may not be possible to obtain such samples. Hence, it is worth investigating the minimum number of samples necessary to obtain good performance in the future. It may also be the case that the number of necessary samples is dependent on the application and dataset being used. Since we are implicitly training the meta-model to capture aleatoric uncertainty, it could be the case that datasets with a large amount of aleatoric uncertainty would require more samples.

6. Conclusions

In this work, we presented a novel UQ framework that consisted of a plug-and-play meta-model, a black-box adversarial attack, and a new soft-label conversion. The meta-model could be easily integrated into machine learning tools. The adversarial attack explored the decision boundary of the base model in a black-box manner and generated more meaningful adversarial samples to alleviate the issue of an imbalance set. The soft-label approach enabled the meta-model to distinguish between low-confidence and misclassified samples. The proposed approach could be applied to any meta-model-based UQ and addresses two fundamental shortcomings in their typical training procedure. We validated the effectiveness of our approach on two base models using two datasets. The soft-label approach significantly improved performance, while the training framework with adversaries could mitigate issues caused by imbalanced training sets. Additionally, we achieved superior or competitive performance according to several widely used and well-known metrics. We also presented several shortcomings of our method and provided interesting future research directions.

Author Contributions: Conceptualization, K.L., A.V. and M.X.; methodology, K.L., A.V. and M.X.; software, K.L.; validation, K.L., A.V. and M.X.; formal analysis, K.L.; investigation, K.L. and M.X.; resources, M.X.; data curation, K.L. and A.V.; writing—original draft preparation, K.L.; writing—review and editing, M.X. and A.V.; visualization, K.L.; supervision, M.X. and A.V.; project administration, M.X. and A.V.; funding acquisition, M.X. and A.V. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Publicly archived datasets. The SVHN dataset [57] can be found http://ufldl.stanford.edu/housenumbers/ (accessed on 11 November 2024). The CIFAR10 dataset [58] can be found https://www.cs.toronto.edu/~kriz/cifar.html (accessed on 11 November 2024).

Acknowledgments: We would like to acknowledge the support from the P3R1 program of University of Idaho.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Zhou, W.; Yang, Y.; Yu, C.; Liu, J.; Duan, X.; Weng, Z.; Chen, D.; Liang, Q.; Fang, Q.; Zhou, J.; et al. Ensembled deep learning model outperforms human experts in diagnosing biliary atresia from sonographic gallbladder images. *Nat. Commun.* **2021**, *12*, 1259. [CrossRef]
2. He, K.; Zhang, X.; Ren, S.; Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1026–1034.
3. Geirhos, R.; Temme, C.R.M.; Rauber, J.; Schütt, H.H.; Bethge, M.; Wichmann, F.A. Generalisation in humans and deep neural networks. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, Montréal, QC, Canada, 3–8 December 2018; Advances in Neural Information Processing; Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2018; Volume 31.
4. Mukhoti, J.; Kulharia, V.; Sanyal, A.; Golodetz, S.; Torr, P.; Dokania, P. Calibrating deep neural networks using focal loss. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 15288–15299.
5. Chen, T.; Navratil, J.; Iyengar, V.; Shanmugam, K. Confidence Scoring Using Whitebox Meta-models with Linear Classifier Probes. In Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics, Naha, Okinawa, Japan, 16–18 April 2019; Proceedings of Machine Learning Research; Volume 89, pp. 1467–1475.
6. Aigrain, J.; Detyniecki, M. Detecting adversarial examples and other misclassifications in neural networks by introspection. In Proceedings of the International Conference on Machine Learning Workshop on Uncertainty and Robustness in Deep Learning, Long Beach, CA, USA, 9–15 June 2019.
7. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
8. Blundell, C.; Cornebise, J.; Kavukcuoglu, K.; Wierstra, D. Weight uncertainty in neural networks. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 1613–1622.
9. Gal, Y.; Ghahramani, Z. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In Proceedings of the 33rd International Conference on Machine Learning, New York, NY, USA, 20–22 June 2016; Proceedings of Machine Learning Research; Volume 48, pp. 1050–1059.
10. Lakshminarayanan, B.; Pritzel, A.; Blundell, C. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. In Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017; Advances in Neural Information Processing Systems; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30.
11. He, B.; Lakshminarayanan, B.; Teh, Y.W. Bayesian deep ensembles via the neural tangent kernel. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 1010–1022.
12. Wilson, A.G.; Izmailov, P. Bayesian deep learning and a probabilistic perspective of generalization. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 4697–4708.
13. Durasov, N.; Bagautdinov, T.; Baque, P.; Fua, P. Masksembles for uncertainty estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 13539–13548.
14. Jain, S.; Liu, G.; Mueller, J.; Gifford, D. Maximizing Overall Diversity for Improved Uncertainty Estimates in Deep Ensembles. *Proc. AAAI Conf. Artif. Intell.* **2020**, *34*, 4264–4271. [CrossRef]
15. Stoyanova, Y.; Ghandi, S.; Tavakol, M. Toward Robust Uncertainty Estimation with Random Activation Functions. *Proc. AAAI Conf. Artif. Intell.* **2023**, *37*, 15152–15160. [CrossRef]

16. Wen, Y.; Tran, D.; Ba, J. BatchEnsemble: An Alternative Approach to Efficient Ensemble and Lifelong Learning. In Proceedings of the International Conference on Learning Representations, Online, 26 April–1 May 2020.
17. Laurent, O.; Lafage, A.; Tartaglione, E.; Daniel, G.; Marc Martinez, J.; Bursuc, A.; Franchi, G. Packed Ensembles for efficient uncertainty estimation. In Proceedings of the Eleventh International Conference on Learning Representations, Kigali, Rwanda, 1–5 May 2023.
18. Havasi, M.; Jenatton, R.; Fort, S.; Liu, J.Z.; Snoek, J.; Lakshminarayanan, B.; Dai, A.M.; Tran, D. Training independent subnetworks for robust prediction. In Proceedings of the International Conference on Learning Representations, Vienna, Austria, 4 May 2021.
19. Van Amersfoort, J.; Smith, L.; Teh, Y.W.; Gal, Y. Uncertainty estimation using a single deep deterministic neural network. In Proceedings of the International Conference on Machine Learning, Online, 13–18 July 2020; pp. 9690–9700.
20. Mukhoti, J.; Kirsch, A.; van Amersfoort, J.; Torr, P.H.; Gal, Y. Deep Deterministic Uncertainty: A New Simple Baseline. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Los Alamitos, CA, USA, 17–24 June 2023; pp. 24384–24394.
21. Liu, J.; Lin, Z.; Padhy, S.; Tran, D.; Bedrax Weiss, T.; Lakshminarayanan, B. Simple and principled uncertainty estimation with deterministic deep learning via distance awareness. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 7498–7512.
22. van Amersfoort, J.; Smith, L.; Jesson, A.; Key, O.; Gal, Y. On Feature Collapse and Deep Kernel Learning for Single Forward Pass Uncertainty. In Proceedings of the Thirty-Fifth Conference on Neural Information Processing Systems Bayesian Deep Learning Workshop, Online, 6–14 December 2021.
23. Miyato, T.; Kataoka, T.; Koyama, M.; Yoshida, Y. Spectral Normalization for Generative Adversarial Networks. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–5 May 2018.
24. Gulrajani, I.; Ahmed, F.; Arjovsky, M.; Dumoulin, V.; Courville, A.C. Improved training of wasserstein gans. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 5769–5779.
25. Arjovsky, M.; Chintala, S.; Bottou, L. Wasserstein Generative Adversarial Networks. In Proceedings of the 34th International Conference on Machine Learning, PMLR, Sydney, Australia, 6–11 August 2017; Proceedings of Machine Learning Research; Volume 70, pp. 214–223.
26. Sehwag, V.; Chiang, M.; Mittal, P. SSD: A Unified Framework for Self-Supervised Outlier Detection. In Proceedings of the International Conference on Learning Representations, Vienna, Austria, 4 May 2021.
27. Mahalanobis, P. On the Generalised Distance in Statistics (Reprint, 2018). *Sankhya A* **1936**, *80*, S1–S7.
28. Corbière, C.; THOME, N.; Bar-Hen, A.; Cord, M.; Pérez, P. Addressing Failure Prediction by Learning Model Confidence. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; Curran Associates, Inc.: Red Hook, NY, USA, 2019; Volume 32.
29. Shen, M.; Bu, Y.; Sattigeri, P.; Ghosh, S.; Das, S.; Wornell, G. Post-hoc Uncertainty Learning Using a Dirichlet Meta-Model. *Proc. AAAI Conf. Artif. Intell.* **2023**, *37*, 9772–9781. [CrossRef]
30. Alain, G.; Bengio, Y. Understanding intermediate layers using linear classifier probes. In Proceedings of the International Conference on Learning Representations, Toulon, France, 24–26 April 2017.
31. Friedman, J.H. Greedy function approximation: A gradient boosting machine. *Ann. Stat.* **2001**, *29*, 1189–1232. [CrossRef]
32. Joo, T.; Chung, U.; Seo, M.G. Being Bayesian about Categorical Probability. In Proceedings of the 37th International Conference on Machine Learning, Online 13–18 July 2020; Proceedings of Machine Learning Research; Volume 119, pp. 4950–4961.
33. Qiu, X.; Miikkulainen, R. Detecting Misclassification Errors in Neural Networks with a Gaussian Process Model. *Proc. AAAI Conf. Artif. Intell.* **2022**, *36*, 8017–8027. [CrossRef]
34. Williams, C.; Rasmussen, C. Gaussian processes for regression. *Adv. Neural Inf. Process. Syst.* **1995**, *8*, *514–520* .
35. Hendrycks, D.; Gimpel, K. A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks. In Proceedings of the International Conference on Learning Representations, Toulon, France, 24–26 April 2017.
36. Dablain, D.; Krawczyk, B.; Chawla, N.V. DeepSMOTE: Fusing Deep Learning and SMOTE for Imbalanced Data. *IEEE Trans. Neural Netw. Learn. Syst.* **2023**, *34*, 6390–6404. [CrossRef] [PubMed]
37. Sahoo, A.; Singh, A.; Panda, R.; Feris, R.; Das, A. Mitigating Dataset Imbalance via Joint Generation and Classification. In Proceedings of the ECCV Workshop on Imbalance Problems in Computer Vision, Glasgow, UK, 23–28 August 2020.
38. Qu, H.; Li, Y.; Foo, L.G.; Kuen, J.; Gu, J.; Liu, J. Improving the reliability for confidence estimation. In Proceedings of the European Conference on Computer Vision, Tel Aviv, Israel, 23–27 October 2022; Springer: Cham, Switzerland, 2022; pp. 391–408.
39. Luo, Y.; Wong, Y.; Kankanhalli, M.S.; Zhao, Q. Learning to predict trustworthiness with steep slope loss. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 21533–21544.
40. Lee, K.; Lee, H.; Lee, K.; Shin, J. Training Confidence-calibrated Classifiers for Detecting Out-of-Distribution Samples. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–5 May 2018.
41. Oberdiek, P.; Fink, G.A.; Rottmann, M. UQGAN: A Unified Model for Uncertainty Quantification of Deep Classifiers trained via Conditional GANs. In Proceedings of the Advances in Neural Information Processing Systems, New Orleans, LA, USA, 28 November–9 December 2022; Curran Associates, Inc.: Red Hook, NY, USA, 2022; Volume 35, pp. 21371–21385.

42. Tang, K.; Miao, D.; Peng, W.; Wu, J.; Shi, Y.; Gu, Z.; Tian, Z.; Wang, W. CODEs: Chamfer Out-of-Distribution Examples Against Overconfidence Issue. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Los Alamitos, CA, USA, 10–17 October 2021; pp. 1153–1162.
43. Hendrycks, D.; Mazeika, M.; Dietterich, T. Deep Anomaly Detection with Outlier Exposure. In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May 2019.
44. Wang, Q.; Ye, J.; Liu, F.; Dai, Q.; Kalander, M.; Liu, T.; HAO, J.; Han, B. Out-of-distribution Detection with Implicit Outlier Transformation. In Proceedings of the Eleventh International Conference on Learning Representations, Kigali, Rwanda, 1–5 May 2023.
45. Arjovsky, M.; Bottou, L. Towards Principled Methods for Training Generative Adversarial Networks. In Proceedings of the International Conference on Learning Representations, Toulon, France, 24–26 April 2017.
46. Yang, J.; Wang, H.; Feng, L.; Yan, X.; Zheng, H.; Zhang, W.; Liu, Z. Semantically Coherent Out-of-Distribution Detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Los Alamitos, CA, USA, 10–17 October 2021; pp. 8301–8309.
47. Du, X.; Wang, Z.; Cai, M.; Li, Y. VOS: Learning What You Don't Know by Virtual Outlier Synthesis. In Proceedings of the International Conference on Learning Representations, Virtual, 25–29 April 2022.
48. Pinto, F.; Yang, H.; Lim, S.N.; Torr, P.; Dokania, P. Using Mixup as a Regularizer Can Surprisingly Improve Accuracy & Out-of-Distribution Robustness. In Proceedings of the Advances in Neural Information Processing Systems, New Orleans, LA, USA, 28 November–9 December 2022; Curran Associates, Inc.: Red Hook, NY, USA, 2022; Volume 35, pp. 14608–14622.
49. Sun, Y.; Ming, Y.; Zhu, X.; Li, Y. Out-of-distribution Detection with Deep Nearest Neighbors. In Proceedings of the 39th International Conference on Machine Learning, Baltimore, MD, USA, 17–23 July 2022; Volume 162 pp. 20827–20840.
50. Kotelevskii, N.; Artemenkov, A.; Fedyanin, K.; Noskov, F.; Fishkov, A.; Shelmanov, A.; Vazhentsev, A.; Petiushko, A.; Panov, M. Nonparametric uncertainty quantification for single deterministic neural network. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 36308–36323.
51. Northcutt, C.G.; Athalye, A.; Mueller, J. Pervasive Label Errors in Test Sets Destabilize Machine Learning Benchmarks. In Proceedings of the 35th Conference on Neural Information Processing Systems Track on Datasets and Benchmarks, Virtual Event, 6–16 December 2021.
52. Vădineanu, Ş.; Pelt, D.M.; Dzyubachyk, O.; Batenburg, K.J. An analysis of the impact of annotation errors on the accuracy of deep learning for cell segmentation. In Proceedings of the International Conference on Medical Imaging with Deep Learning, PMLR, Zurich, Switzerland, 6–8 July 2022; pp. 1251–1267.
53. Hinton, G.; Vinyals, O.; Dean, J. Distilling the knowledge in a neural network. In Proceedings of the The Twenty-Eighth Annual Conference on Neural Information Processing Systems Deep Learning Workshop, Montréal, QC, Canada, 8–13 December 2014.
54. Goodfellow, I.J.; Shlens, J.; Szegedy, C. Explaining and harnessing adversarial examples. In Proceedings of the International Conference on Learning Representations, San Diego, CA, USA, 7–9 May 2015.
55. Chen, J.; Gu, Q. RayS: A Ray Searching Method for Hard-label Adversarial Attack. In Proceedings of the 26rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Virtual Event, 6–10 July 2020.
56. Ma, Y.; Lucke, K.; Xian, M.; Vakanski, A. Semantic-Aware Adaptive Binary Search for Hard-Label Black-Box Attack. *Computers* **2024**, *13*, 203. [CrossRef]
57. Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; Ng, A.Y. Reading digits in natural images with unsupervised feature learning. In Proceedings of the NIPS Workshop on Deep Learning and Unsupervised Feature Learning, Granada, Spain, 16 December 2011; p. 4.
58. Krizhevsky, A.; Hinton, G. *Learning Multiple Layers of Features from Tiny Images*; Technical Report; University of Toronto: Toronto, ON, Canada, 2009.
59. Baeza-Yates, R.; Ribeiro-Neto, B. *Modern Information Retrieval*; Addison Wesley: Boston, MA, USA, 2011; pp. 327–328.
60. Ruder, S. An overview of gradient descent optimization algorithms. *arXiv* **2016**, arXiv:1609.04747.
61. Good, I.J. Rational Decisions. *J. R. Stat. Society. Ser. B (Methodol.)* **1952**, *14*, 107–114. [CrossRef]
62. Kingma, D.P. Adam: A method for stochastic optimization. In Proceedings of the International Conference for Learning Representations, San Diego, CA, USA, 7–9 May 2015.
63. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
64. Glorot, X.; Bordes, A.; Bengio, Y. Deep Sparse Rectifier Neural Networks. In Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA, 11–13 April 2011; Proceedings of Machine Learning Research; Volume 15, pp. 315–323.
65. McNemar, Q. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika* **1947**, *12*, 153–157. [CrossRef]

66. Edwards, A.L. Note on the "correction for continuity" in testing the significance of the difference between correlated proportions. *Psychometrika* **1948**, *13*, 185–187. [CrossRef] [PubMed]
67. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An imperative style, high-performance deep learning library. In Proceedings of the 33rd International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; Curran Associates Inc.: Red Hook, NY, USA, 2019.
68. Raschka, S. MLxtend: Providing machine learning and data science utilities and extensions to Python's scientific computing stack. *J. Open Source Softw.* **2018**, *3*, 638. [CrossRef]
69. Sun, S.; Xian, M.; Vakanski, A.; Ghanem, H. MIRST-DM: Multi-instance RST with drop-max layer for robust classification of breast cancer. In Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, Singapore, 18–22 September 2022; Springer: Cham, Switzerland, 2022; pp. 401–410.
70. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Fei-Fei, L. ImageNet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255. [CrossRef]
71. Cui, Y.; Song, Y.; Sun, C.; Howard, A.; Belongie, S. Large scale fine-grained categorization and domain-specific transfer learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4109–4118.
72. Gupta, K.; Pesquet-Popescu, B.; Kaakai, F.; Pesquet, J.C.; Malliaros, F.D. An adversarial attacker for neural networks in regression problems. In Proceedings of the IJCAI Workshop on Artificial Intelligence Safety (AI Safety), Virtual Event, 19–20 August 2021.
73. Kong, X.; Ge, Z. Adversarial Attacks on Regression Systems via Gradient Optimization. *IEEE Trans. Syst. Man Cybern. Syst.* **2023**, *53*, 7827–7839. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.