DAUX: a Density-based Approach for Uncertainty eXplanations

Preprint · July 2022

DOI: 10.48550/arXiv.2207.05161

5 authors, including:

Boris van Breugel
University of Cambridge
9 PUBLICATIONS **55** CITATIONS
SEE PROFILE

Jonathan Crabbé
University of Cambridge
14 PUBLICATIONS **16** CITATIONS
SEE PROFILE

Nabeel Seedat
29 PUBLICATIONS **54** CITATIONS
SEE PROFILE

DAUX: a Density-based Approach for Uncertainty eXplanations

Hao Sun [*1] **Boris van Breugel** [*1] **Jonathan Crabbé** [1] **Nabeel Seedat** [1] **Mihaela van der Schaar** [1 2 3]

Abstract

Uncertainty quantification (UQ) is essential for creating trustworthy machine learning models. Recent years have seen a steep rise in UQ methods that can flag suspicious examples, however, it is often unclear what exactly these methods identify. In this work, we propose an assumption-light method for interpreting UQ models *themselves*. We introduce the confusion density matrix—a kernel-based approximation of the misclassification density—and use this to categorize suspicious examples identified by a given UQ method into three classes: out-of-distribution (OOD) examples, boundary (Bnd) examples, and examples in regions of high in-distribution misclassification (IDM). Through extensive experiments, we shed light on existing UQ methods and show that the cause of the uncertainty differs across models. Additionally, we show how the proposed framework can make use of the categorized examples to improve predictive performance. [1]

1. Introduction

Although black-box parametric models like neural networks have achieved remarkably good performance on a variety of challenging tasks, many real-world applications with safety concerns like healthcare (Aggarwal et al., 2021), finance (Kim et al., 2020; Ding et al., 2020) and autonomous driving (Kim & Canny, 2017; Sun et al., 2021) necessitate reliability and explainability. Issued predictions in those scenarios must be trustworthy to avoid high cost of erroneous decisions.

Uncertainty Quantification (UQ) (Lakshminarayanan et al., 2017; Blundell et al., 2015; Graves, 2011; Ghosh et al., 2018; Gal & Ghahramani, 2016) addresses the challenge of trustworthy prediction through inspecting the confidence of a model, enabling intervention whenever uncertainty is too high. The cause of the uncertainty, however, is not explained. In this work, we go beyond black-box UQ methods and aim to answer two questions:

1. How do we provide a more granular explanation of *why* uncertainty methods identify certain predictions as suspicious?
2. How can these explanations be used to give better predictions for uncertain examples?

Explaining Uncertainty We propose DAUX: a model-agnostic framework that provides post-hoc explanations for model uncertainty. We categorize the model's uncertainty into three classes: (1) **OOD** uncertainty caused by OOD examples, i.e. test-time examples that resemble no training-time sample (Malinin & Gales, 2018; Van Amersfoort et al., 2020; Abdar et al., 2021b; Ran et al., 2022); (2) **Bnd** uncertainty caused by neighboring decision boundaries, i.e., non-conformal predictions due to confusing training-time resemblances from different classes or inherent ambiguities making it challenging to classify the data (Tsipras et al., 2020; Kishida & Nakayama, 2019; Ekambaram et al., 2017); (3) **IDM** uncertainty caused by imperfections in the model—as manifested by high misclassification at validation time—such that similar misclassification is to be expected during testing (Ramalho & Miranda, 2020). Figure 1 illustrates the different classes and in Section 5 we provide visualisations using real data. We show how DAUX can be integrated with a broad class of UQ methods to understand suspicious misclassifications, in pursuance of reliable predictions for real-world deployments.

Improving prediction We explore how DAUX's fine-grained uncertainty explanations can inform the training of additional models, for the purpose of increasing prediction performance on uncertain test-time examples. We call this the inverse direction (cf. the forward direction of explaining uncertainty). Specifically, based on the key insight that test-time uncertainty can be explained by validation-time misclassification of similar data, we propose to improve uncertain predictions by adapting the training set according to DAUX's uncertainty explanations.

[*]Equal contribution [1]Department of Applied Mathematics and Theoretical Physics, University of Cambridge, UK [2]The Alan Turing Institute, London, UK [3]University of California, Los Angeles, USA. Correspondence to: Hao Sun <hs789@cam.ac.uk>.

Published at the DFUQ Workshop at the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

[1]Code is available at https://anonymous.4open.science/r/DAUX-CBBF

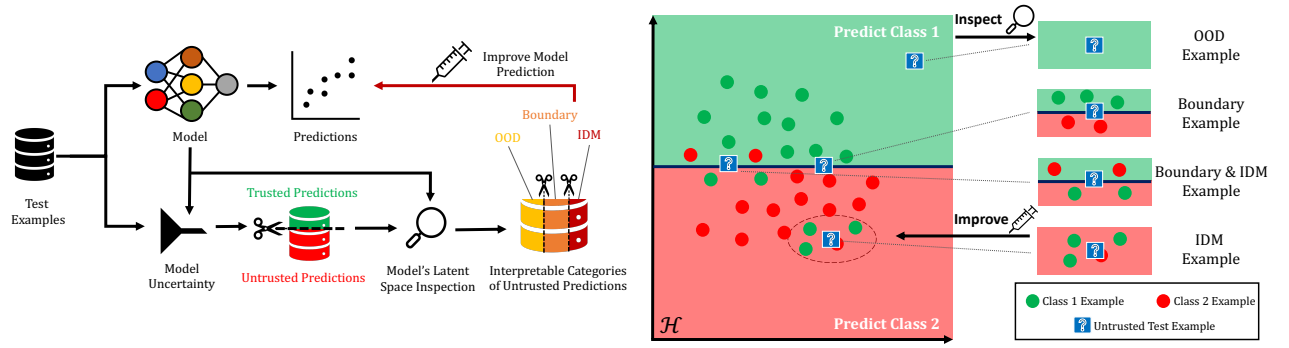


Figure 1. Given a prediction and UQ algorithm, our method divides flagged test time examples into four classes. Class **OOD** identifies outliers, which are mistrusted because they do not resemble training data; class **IDM** indicates examples lying in regions with high misclassification; class **Bnd** indicates examples that lie near the decision boundary.

Our contributions can be summarized as follows:

1. We propose the confusion density matrix—the heart of DAUX—which allows us to to offer an explanatory characterisation of uncertain examples at test time.
2. Empirically, we use DAUX to compare existing UQ methods. By explaining the different methods' sensitivity to different types of uncertainty, this provides model insight and aid UQ method selection.
3. We demonstrate how we can improve the classification performance by fine-tuning the classifier based on DAUX's explanations.

2. Related Work

Uncertainty Quantification UQ methods are used to assess the confidence in a model's predictions. In recent years the machine learning community have proposed many methods which broadly fall into the following categories: (1) Ensemble methods (i.e. Deep Ensembles (Lakshminarayanan et al., 2017)), which while considered state of the art have a high computational burden; (2) Approximate Bayesian methods (i.e. Stochastic Variational Inference (Blundell et al., 2015; Graves, 2011; Ghosh et al., 2018)), however work by (Yao et al., 2019; Ovadia et al., 2019; Foong et al., 2019) suggests these methods do not yield high quality uncertainty estimates; and (3) Dropout based methods such as Monte Carlo Dropout (Gal & Ghahramani, 2016), which are simpler as they do rely on estimating a posterior, however the quality of the uncertainty estimates is linked to the choice of parameters which need to be calibrated to match the level of uncertainty and avoid sub-optimal performance (Verdoja & Kyrki, 2020). For completeness we note that whilst, conformal prediction (Vovk et al., 2005) is another UQ method, the paradigm is different from the other aforementioned methods, as it returns predictive sets to satisfy coverage guarantees rather than a value based measure of uncertainty.

In practice, the predictive uncertainty for a model prediction arises from the lack of relevant training data (epistemic uncertainty) or the inherent non-separable property of the data distribution (aleatoric uncertainty) (Kendall & Gal, 2017; Abdar et al., 2021a; Hüllermeier & Waegeman, 2021). This distinction in types of uncertainty is crucial as it has been shown that samples with low epistemic uncertainty are more likely under the data distribution, hence motivating why epistemic uncertainty has been used for OOD/outlier detection (Gal & Ghahramani, 2016). Moreover, ambiguous instances close to the decision boundary typically result in high aleatoric uncertainty (Schut et al., 2021).

This suggests that different sets of uncertain data points are associated with different types of uncertainties and as a consequence different types of misclassifications. Furthermore, it is to be expected that different uncertainty estimators are more able/prone to capturing some types of uncertainty than others. To understand these models better, we thus require a more granular definition to characterize uncertain data points. We employ three classes: outliers (OOD), boundary examples (Bnd) and examples of regions with high in-distribution misclassification (IDM), see Fig 1.

Out-of-distribution (OOD) detection Recall that UQ methods have been used to flag OOD examples. For completeness, we highlight that other alternative methods exist for OOD detection. For example, (Lee et al., 2018) detected OOD examples based on the Mahalanobis distance, whilst (Ren et al., 2019) use the likelihood ratio between two generative models. We emphasise that DAUX's aim is broader—creating a unifying framework for explaining multiple types of uncertainty—however existing OOD methods could be used to replace DAUX's OOD detector.

Accuracy without Labels We contrast our work to the literature which aims to determine model accuracy without access to ground-truth labels. Methods such as Ghanta et al. (2019); Deng & Zheng (2021) propose a secondary regression model as an accuracy predictor given a data sample.

Table 1. Comparison with related work in UQ. We aim to provide a flexible, interpretable framework for understanding mistrusted examples identified by uncertainty estimators. Furthermore, this framework enables us to improves the prediction performance on a certain type of flagged uncertain class.

Method	Model Structure	Uncertainty Estimation	Explain Uncertainty	Improve Prediction	Examples
BNNs	Bayesian Layers	✓	·	·	Graves (2011); Ghosh et al. (2018)
GP	Gaussian Processes	✓	·	·	Rasmussen (2003)
MC-Dropout	Drop-Out Layers	✓	·	·	Gal & Ghahramani (2016)
Deep-Ensemble	Multiple Models	✓	·	·	Lakshminarayanan et al. (2017)
Conformal Prediction	Model "Wrapper"	✓	·	·	Vovk et al. (2005); Balasubramanian et al. (2014),
Performance Prediction	Multiple Predictive Models	✓	·	·	Ghanta et al. (2019); Ramalho & Miranda (2020)
DAUX	Assumption 1	✓	✓	✓	**(Ours)**

Ramalho & Miranda (2020) combines regression model with latent nearest neighbors for uncertain prediction. This is different from our setting which is focused on understanding UQ methods on a sample level by characterizing it as an outlier, boundary or IDM example.

3. Forward Direction: Understand Model Uncertainty via Latent Density

3.1. Preliminaries

We consider a typical classification setting where $\mathcal{X} \subseteq \mathbb{R}^{d_X}$ is the input space and $\mathcal{Y} = [0,1]^C$ is the set of class probabilities, where d_X is the dimension of input and output space and $C \in \mathbb{N}^*$ is the number of classes. We are given a prediction model $f : \mathcal{X} \to \mathcal{Y}$ that maps $x \in \mathcal{X}$ to class probabilities $f(x) \in \mathcal{Y}$. An uncertainty estimator $u : \mathcal{X} \to [0,1]$ quantifies the uncertainty of the issued predictions. Given some threshold τ, the inference-time predictions can be separated into trusted predictions $\{x \in \mathcal{X} | u(x) < \tau\}$ and untrusted predictions $\{x \in \mathcal{X} | u(x) \geq \tau\}$. We make the following assumption on the model architecture of f.

Assumption 1 (Model Architecture)**.** *Model f can be decomposed as $f = \varphi \circ l \circ g$, where $g : \mathcal{X} \to \mathcal{H} \subseteq \mathbb{R}^{d_H}$ is a feature extractor that maps the input space to a $d_H < d_X$ dimensional latent (or representation) space, $l : \mathcal{H} \to \mathbb{R}^C$ is a linear map between the latent space to the output space and $\varphi : \mathbb{R}^C \to \mathcal{Y}$ is a normalizing map that converts vectors into probabilities.*

Remark 1. *This assumption guarantees that the model is endowed with a lower-dimensional representation space. Most modern UQ methods like MCD, Deep-Ensemble, BNN satisfy this assumption. In the following, we use this representation space to explain model uncertainty.*

We assume that the model and the uncertainty estimator have been trained with a set of $N \in \mathbb{N}^*$ training examples $\mathcal{D}_{\text{train}} = \{(x^n, y^n) \mid n \in [N]\}$. At inference time the underlying model f and uncertainty method u predict class probabilities $f(x) \in \mathcal{Y}$ and uncertainty $u(x) \in [0,1]$, respectively. We assign a class to this probability vector $f(x)$ with the map $\text{class} : \mathcal{Y} \to [C]$ that maps a probability vector y to the class with maximal probability $\text{class}[y] = \arg\max_{c \in [C]} y_c$. While uncertainty estimators flag examples to be trustworthy or not, they do not provide a reason for why a certain prediction is to be mistrusted. Our aim is to use the model's predictions and representations of a corpus of labelled examples—which we will usually take to be the training ($\mathcal{D}_{\text{train}}$) or validation ($\mathcal{D}_{\text{val}}$) sets—to understand inference-time uncertainty predictions. To that aim, we distinguish two general scenarios where a model's predictions should be considered with skepticism.

3.2. Flagging OOD Examples

There is a limit to model generalization. Uncertainty estimators should be skeptical when the input $x \in \mathcal{X}$ differs significantly from input examples that the model was trained on. From an UQ perspective, the predictions for these examples are expected to be associated with a large epistemic uncertainty.

A natural approach to flagging these examples is to define a density $p(\cdot \mid \mathcal{D}_{\text{train}}) : \mathcal{X} \to \mathbb{R}^+$ over the input space. This density should be such that $p(x \mid \mathcal{D}_{\text{train}})$ is high whenever the example $x \in \mathcal{X}$ resembles one or several examples from the training set $\mathcal{D}_{\text{train}}$. Conversely, a low value for $p(x \mid \mathcal{D}_{\text{train}})$ indicates that the example x differs from the training examples. Of course, estimating the density $p(x \mid \mathcal{D}_{\text{train}})$ is a nontrivial task. At this stage, it is worth noting that this density does not need to reflect the ground-truth data generating process underlying the training set $\mathcal{D}_{\text{train}}$. For the problem at hand, this density $p(x \mid \mathcal{D}_{\text{train}})$ need only measure how close the example x is to the training data manifold. A common approach is to build a kernel density estimation with the training set $\mathcal{D}_{\text{train}}$. Further, we note that Assumption 1 provides a representation space $\mathcal{H}$ that was specifically learned for the classification task on $\mathcal{D}_{\text{train}}$. In Appendix A.1, we argue that this latent space is suitable for our kernel density estimation. This motivates the following definition for $p(\cdot \mid \mathcal{D}_{\text{train}})$.

Table 2. Summary of different uncertainty types

Type	Definition	Description
OOD	$1/p(x\|\mathcal{D}_{\text{train}}) > \tau_{\text{OOD}}$	Samples that do not resemble the training data. Additional labelled data that covers this part of the input space is required to improve performance on these samples.
Bnd	$\sum_{c\neq\hat{c}} P_{c,c}(x\|\mathcal{D}_{\text{val}}) > \tau_{\text{Bnd}}$	Samples near the boundaries in the latent space. Predictions on these samples are sensitive to small changes in the predictor, and fine-tuning the prediction model may yield better predictions.
IDM	$\sum_{c\neq\hat{c}} P_{c,\hat{c}}(x\|\mathcal{D}_{\text{val}}) > \tau_{\text{IDM}}$	Samples that are likely to be misclassified, since similar examples were misclassified in the validation set. For samples that are also flagged as Bnd, finetuning of the current prediction model could yield better predictions. For samples that are not also flagged as Bnd, better predictions may be possible, but this requires training a different model for the latent representation.

Definition 1 (Latent Density)**.** *Let $f : \mathcal{X} \to \mathcal{Y}$ be a prediction model, let $g : \mathcal{X} \to \mathcal{H}$ be the feature extractor from Assumption 1 and let $\kappa : \mathcal{H} \times \mathcal{H} \to \mathbb{R}^+$ be a kernel function. The* latent density $p(\cdot \mid \mathcal{D}) : \mathcal{X} \to \mathbb{R}^+$ *is defined over a dataset $\mathcal{D}$ as:*

$$p(x \mid \mathcal{D}) \equiv \frac{1}{N} \sum_{\tilde{x}\in\mathcal{D}} \kappa\left[g(x), g(\tilde{x})\right] \tag{1}$$

Test examples with a low training density are likely to be underfitted by the model, and thus not to be trusted.

Definition 2 (OOD Score)**.** *The* OOD Score T_{OOD} *is defined as*

$$T_{\text{OOD}}(x) \equiv \frac{1}{p(x|\mathcal{D}_{\text{train}})} \tag{2}$$

For a test example $x \in \mathcal{D}_{\text{test}}$, if $T_{\text{OOD}}(x|\mathcal{D}_{\text{train}}) \geq \tau_{\text{OOD}}$ the example is suspected to be an outlier with respect to the training set. We set $\tau_{\text{OOD}} = \frac{1}{\min_{x'\in\mathcal{D}_{\text{train}}} p(x'|\mathcal{D}_{\text{train}})}$, i.e. a new sample's training density is smaller than the minimal density of training examples.

3.3. Flagging IDM and Boundary Examples

Samples that are not considered outliers, yet are given high uncertainty scores, we divide up further into two non-exclusionary category. The first category consists of points located near the boundary between two or more classes, the second consists of points that are located in regions of high misclassification.

For achieving this categorisation, we will use a separate validation set $\mathcal{D}_{\text{val}}$. We first partition the validation examples according to their true and predicted label. More precisely, for each couple of classes $(c_1, c_2) \in [C]^2$, we define the corpus $\mathcal{C}_{c_1\mapsto c_2} \equiv \{(x, y) \in \mathcal{D}_{\text{val}} \mid \text{class}[y] = c_1 \wedge \text{class}[f(x)] = c_2\}$ of validation examples whose true class is c_1 and whose predicted class is c_2. In the case where these two classes are different $c_1 \neq c_2$, this corresponds to a corpus of misclassified examples. Clearly, if some example $x \in \mathcal{X}$ resembles one or several examples of those misclassification corpus, it is legitimate to be sceptical about the prediction $f(x)$ for this example. In fact, keeping track of the various misclassification corpora from the validation set allows to have an idea on what the misclassification is likely to be.

In order to make this detection of suspicious examples quantitative, we will mirror the approach from Definition 1. Indeed, we can define a kernel density $p(\cdot \mid \mathcal{C}_{c_1\mapsto c_2}) : \mathcal{X} \to \mathbb{R}^+$ for each corpus $\mathcal{C}_{c_1\mapsto c_2}$. Again, this density will be such that $p(\cdot \mid \mathcal{C}_{c_1\mapsto c_2})$ is high whenever the representation of the example $x \in \mathcal{X}$ resembles the representation of one or several examples from the corpus $\mathcal{C}_{c_1\mapsto c_2}$. If this corpus is a corpus of misclassified examples, this should trigger our scepticism about the model's prediction $f(x)$. By aggregating the densities associated to each of these corpora, we arrive at the following definition.

Definition 3 (Confusion Density Matrix)**.** *Let $f : \mathcal{X} \to \mathcal{Y}$ be a prediction model, let $g : \mathcal{X} \to \mathcal{H}$ be the feature extractor from Assumption 1 and let $\kappa : \mathcal{H} \times \mathcal{H} \to \mathbb{R}^+$ be a kernel function. The* confusion density matrix $P(\cdot \mid \mathcal{D}_{\text{val}}) : \mathcal{X} \to (\mathbb{R}^+)^{C\times C}$ *is defined as*

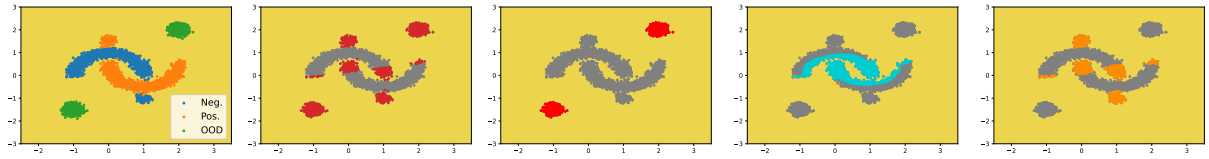
$$\begin{aligned} P_{c_1,c_2}(x \mid \mathcal{D}_{\text{val}}) &\equiv p(x \mid \mathcal{C}_{c_1\mapsto c_2}) \\ &= \frac{1}{|\mathcal{C}_{c_1\mapsto c_2}|} \sum_{\tilde{x}\in\mathcal{C}_{c_1\mapsto c_2}} \kappa[g(x), g(\tilde{x})] \end{aligned} \tag{3}$$

for all $(c_1, c_2) \in [C]^2$.

Remark 2. *The name* confusion density *is chosen to make a parallel with confusion matrices. Like confusion matrices, our confusion density indicates the likelihood of each couple (c_1, c_2), where c_1 is the true class and c_2 is the predicted class. Unlike confusion matrices, our confusion density provides an instance-wise (i.e. for each $x \in \mathcal{X}$) likelihood for each couple.*

3.3.1. BND EXAMPLES

By inspecting the confusion matrix, we can quantitatively distinguish two situations where the example $x \in \mathcal{X}$ is likely to be mistrusted. The first situation where high uncertainty arises, is when $g(x)$ is close to latent representations of validation examples that have been correctly assigned a label that differs from the predicted one $\text{class}[f(x)]$. This typically happens when $g(x)$ is located close to a decision boundary in latent space. In our validation confusion matrix,



(a) Test Dataset (b) Test Errors (c) Flagged OOD (d) Flagged Bnd (e) Flagged IDM

Figure 2. Visualization of our proposed framework on the Two-Smiles dataset with a linear model.

this likelihood that x is related to validation examples with different labels is reflected by the diagonal elements. This motivates the following definition.

Definition 4 (Boundary Score)**.** *Let* $P(x \mid \mathcal{D}_{\text{val}})$ *be the* confusion density matrix *for an example* $x \in \mathcal{X}$ *with predicted class* $\hat{c} = \text{class}[f(x)]$*. We define the* boundary score *as the sum of each density of well-classified examples from a different class:*

$$T_{\text{Bnd}}(x) = \sum_{c \neq \hat{c}}^{C} P_{c,c}\left(x \mid \mathcal{D}_{\text{val}}\right). \tag{4}$$

Points are identified as Bnd when $T_{\text{Bnd}} > \tau_{\text{Bnd}}$—see Appendix A.2.

3.3.2. IDM EXAMPLES

The second situation is the one previously mentioned: the latent representation $g(x)$ is close to latent representations of validation examples that have been misclassified. In our validation confusion matrix, this likelihood that x is related to misclassified validation examples is reflected by the off-diagonal elements. This motivates the following definition.

Definition 5 (IDM Score)**.** *Let* $P(x \mid \mathcal{D}_{\text{val}})$ *be the* confusion density matrix *for an example* $x \in \mathcal{X}$*. We define the* IDM score *as the sum of each density corresponding to a misclassification of the predicted class* $\hat{c} = \text{class}\, f(x)$ *in the confusion density matrix:*

$$T_{\text{IDM}}(x) = \sum_{c \neq \hat{c}}^{C} P_{c,\hat{c}}\left(x \mid \mathcal{D}_{\text{val}}\right). \tag{5}$$

Points are identified as IDM when $T_{\text{IDM}} > \tau_{\text{IDM}}$. We choose τ_{IDM} such that the proportion of IDM points in the validation set equals the number of misclassified examples. Details for definitions and choices of thresholds are provided in Appendix A.2.

Remark 3. *Note that the definitions of Bnd examples and IDM examples do not exclude each other, therefore, an uncertain example can be flagged as a Bnd example, an IDM example, or flagged as both Bnd and IDM (B&I). To make this distinction clear, we will refer to the disjoint classes as:*

$$\mathcal{S}_{\text{Bnd}} = \{x | x \in \mathcal{D}_{\text{test}}, T_{\text{Bnd}}(x) > \tau_{\text{Bnd}}, T_{\text{IDM}}(x) \leq \tau_{\text{IDM}}\}$$
$$\mathcal{S}_{\text{IDM}} = \{x | x \in \mathcal{D}_{\text{test}}, T_{\text{Bnd}}(x) \leq \tau_{\text{Bnd}}, T_{\text{IDM}}(x) > \tau_{\text{IDM}}\}$$
$$\mathcal{S}_{\text{B\&I}} = \{x | x \in \mathcal{D}_{\text{test}}, T_{\text{Bnd}}(x) > \tau_{\text{Bnd}}, T_{\text{IDM}}(x) > \tau_{\text{IDM}}\}$$

Test examples that are flagged by uncertainty method u—yet do not meet any of the thresholds—are marked as *Other*.

In a nutshell DAUX uses the OOD, IDM and Bnd classes to explain model uncertainty—see Table 2 for an overview. Better predictions may be possible for IDM samples, in case a different model is used. For samples that are also labelled as Bnd, fine-tuning the existing model—possibly after gathering more data—may be sufficient to separate the different classes better. IDM samples that are not in the Bnd class are harder, and may only be classified correctly if a different latent representation is found. In Section 5.1 we explore the distinction between classes further.

4. Inverse Direction: Improving Uncertain Predictions

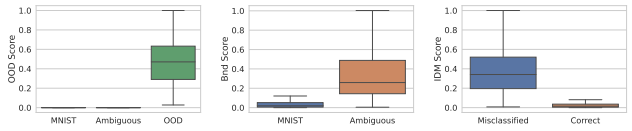
Knowing the category that a suspicious example belongs to, can we improve its prediction? For ease of exposition, we focus on improving predictions for $\mathcal{S}_{\text{B\&I}}$.

Let $p\left(x \mid \mathcal{S}_{\text{B\&I}}\right)$ be the latent density be defined as in Definition 1. We can improve the prediction performance of the model on $\mathcal{S}_{\text{B\&I}}$ examples by focusing on the part of examples in the training set that are closely related to those suspicious examples. We propose to refine the training dataset $\mathcal{D}_{\text{train}}$ by only keeping the examples that resembles the latent representations for the specific type of test-time suspicious examples, and train another model on this subset of the training data:

$$\tilde{\mathcal{D}}_{\text{train}} \equiv \{x \in \mathcal{D}_{\text{train}} | p\left(x \mid \mathcal{S}_{\text{B\&I}}\right) \geq \tau_{\text{test}}\}, \tag{6}$$

where τ_{test} is a threshold that can be adjusted to keep a prespecified proportion q of the related training data—see Appendix A.3 for details. Subsequently, new prediction model $f_{\text{B\&I}}$ is trained on $\tilde{\mathcal{D}}_{\text{train}}$.

Orthogonal to ensemble methods that require multiple models trained independently, and improve *overall* prediction accuracy by bagging or boosting, our method is targeted at improving the model's performance on a *specified* subclass of test examples by finding the most relevant training examples. Our method is therefore more transparent and can be used in parallel with ensemble methods if needed.



(a) OOD (b) Bnd (c) IDM

Figure 3. Box-plots of different data classes and corresponding scores. All values are scaled by being divided by the maximal value.

5. Experiments

In this section, we demonstrate our proposed method with empirical studies. In Sec. 5.1 we visualize the different classes of flagged examples on a modified Two-Moons dataset and in Sec. 5.2 we quantitatively assess DAUX's explanation accuracy on the Dirty-MNIST dataset (Mukhoti et al., 2021). Sec. 5.3 presents a use-case of DAUX—comparing existing UQ benchmarks—and in Sec. 5.4 we show how DAUX's explanations can improve uncertain predictions.

5.1. Visualizing DAUX with Two-Smiles

5.1.1. Dataset Description

To highlight the different types of uncertainty, we create a modified version of the Two-Moons dataset, which we will call "Two-Smiles". We use scikit-learn's *datasets* package to generate 6000 two-moons examples with a noise rate of 0.1. In addition, we generate two Gaussian clusters of 150 examples centered at $(0, 1.5)$ and $(1, -1)$ for training, validation and test, and mark them as the positive and negative class separately. The data is split into a training, validation and test set. We add an additional 1500 OOD examples to the test set, that are generated with a Gaussian distribution centered at $(2, 2)$ and $(-1, -1.5)$. Overall, the test set counts 4500 examples, out of which 1500 are positive examples, 1500 are negative examples and 1500 are OOD—see Figure 2 (a).

5.1.2. Results

We demonstrate the intuition behind the different types of uncertainty, by training a linear model to classify the Two-Smiles data—i.e. using a composition of an identity embedding function and linear classifier, see Assumption 1. Figure 2 (b) shows the test-time misclassified examples. Since the identity function does not linearly separate the two classes and outliers are unobserved at training time, the classifier makes mistakes near class boundaries, in predicting the OOD examples, as well as the clusters at $(0, 1.5)$ and $(1, -1)$. In Figure 2 (c)-(e), we see that DAUX correctly flags the OOD examples, boundary examples and the IDM examples. Also note the distinction between $\mathcal{S}_{\text{B\&I}}$ and $\mathcal{S}_{\text{IDM}}$. Slightly fine-tuning the prediction model may yield a decision boundary that separates the $\mathcal{S}_{\text{B\&I}}$ samples from the incorrect class. On the other hand, this will not suffice for the $\mathcal{S}_{\text{IDM}}$ examples—i.e. the clusters at $(0, 1.5)$ and $(1, -1)$—which require a significantly different representation/prediction model to be classified correctly.

Table 3. Quantitative results on the Dirty-MNIST dataset. Our proposed method can flag all three classes of uncertain examples with high F1-Score.

Explanation	Precision	Recall	F1-Score
OOD	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
Bnd	0.959 ± 0.008	0.963 ± 0.006	0.961 ± 0.003
IDM	0.918 ± 0.039	0.932 ± 0.024	0.924 ± 0.012

5.2. Verifying DAUX with Dirty-MNIST

5.2.1. Dataset Description

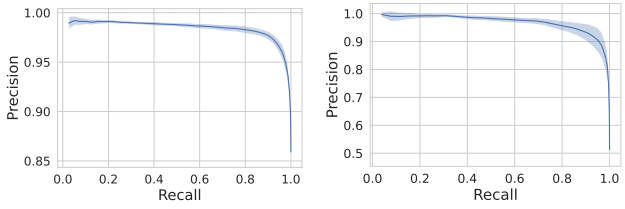
We evaluate the effectiveness of DAUX on the Dirty-MNIST dataset (Mukhoti et al., 2021). The training data set is composed of the vanilla MNIST dataset and Ambiguous-MNIST, containing ambiguous artificial examples—e.g. digits 4 that look like 9s. The test set is similar, but also contains examples from Fashion-MNIST as OOD examples. The advantage of this dataset is that the different data types roughly correspond to the categories we want to detect, and as such we can verify whether DAUX's classification corresponds to these "ground-truth" labels. Specifically, we deem all examples from MNIST to be non-suspicious, associate Ambiguous-MNIST with class Bnd, and associate Fashion-MNIST with class OOD (There are $1,000$ OOD examples out of the $11,000$ test examples). See (Mukhoti et al., 2021) for more details on the dataset.

5.2.2. Results

We use a 3-layer CNN model with ReLU activation and MaxPooling as the backbone model in this section.

Flagging Outliers We assume the ground-truth label of all Fashion-MNIST data is OOD. Comparing these to the classification by DAUX, we are able to quantitatively evaluate the performance of DAUX in explaining outliers. We compute precision, recall and F1-scores for different classes, see Figure 3 (a) and Table 3. All Fashion-MNIST examples are successfully flagged as OOD by DAUX. In Appendix B.3 we include more OOD experiments, including comparison to OOD detector baselines.

Flagging Bnd Examples We expect most of the boundary examples to belong to the Ambiguous-MNIST class, as these have been synthesised using a linear interpolation of two different digits in latent space (Mukhoti et al., 2021).



(a) Flag Bnd Examples (b) Flag IDM Examples

Figure 4. Precision-Recall curves for different choices of thresholds in flagging Bnd examples and IDM examples. Titles above figures are the ground-truth labels.

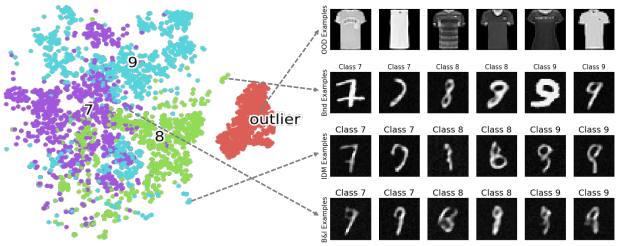


Figure 5. Examples of different uncertainty classes $\mathcal{S}_{\text{OOD}}$, $\mathcal{S}_{\text{IDM}}$, $\mathcal{S}_{\text{Bnd}}$ and $\mathcal{S}_{\text{B\&I}}$. For better visualization, we only plot some classes including the outliers.

Figure 3 (b) shows that DAUX's boundary scores are indeed significantly higher in Ambiguous-MNIST compared to vanilla MNIST. Figure 4 (a) shows the precision-recall curve of DAUX, created by varying threshold τ_{Bnd}. Most boundary examples are correctly explained under a wide range of threshold choices. This stability of uncertainty explanations is desirable, since τ_{Bnd} is usually unknown exactly.

Flagging IDM Examples In order to quantitatively evaluate the performance of DAUX on flagging IDM examples, we use a previously unseen hold-out set, which is balanced to consist of 50% misclassified and 50% correctly classified examples. We label the former as test-time IDM examples and the latter as non-IDM examples, and compare this to DAUX's explanations. Figure 3c shows DAUX successfully assigns significantly higher IDM scores to the examples that are to-be misclassified.

Varying τ_{IDM} we create the precision-recall curve for the IDM class in Figure 4 (b), which is fairly stable w.r.t. the threshold τ_{IDM}. In practice, we recommend to use the prediction accuracy on the validation dataset for setting τ_{IDM}—see Appendix A.2.

Visualizing Different Classes Figure 5 shows examples from the $\mathcal{S}_{\text{OOD}}$, $\mathcal{S}_{\text{Bnd}}$, $\mathcal{S}_{\text{IDM}}$ and $\mathcal{S}_{\text{B\&I}}$ sets. The first row shows OOD examples from the Fashion-MNIST dataset. The second row shows boundary examples, most of which indeed resemble more than one class. The third row shows IDM examples, which DAUX thinks are likely to be misclassified since mistakes were made nearby on the validation set. Indeed, these examples look like they come from the "dirty" part of dirty-MNIST, and most digits are not clearly classifiable. The last row contains B&I examples, which exhibit both traits.

5.3. Explaining Model Uncertainty

In this section, we demonstrate how DAUX explains existing UQ model uncertainty. We compare UQ methods MC-Dropout (Gal & Ghahramani, 2016) (**MCD**), Deep-Ensemble (Lakshminarayanan et al., 2017) (**DE**) and Bayesian Neural Networks (Ghosh et al., 2018) (**BNNs**). These methods output predictions and uncertainty scores simultaneously, we follow the traditional approach to mark examples as uncertain or trusted according to their uncertain scores and specified thresholds. To demonstrate how DAUX explain all ranges of uncertain examples, we present results from top 5% to the least 5% uncertainty.

Figure 6 compares the proportion of different classes of flagged examples across the three UQ methods. The first and second rows show the *total number* and *proportion* of flagged examples for each class, respectively. There is a significant difference in what the different UQ methods identify. There is a significant difference between types of classes that the different UQ methods identify as discussed in the literature (Yao et al., 2019; Ovadia et al., 2019; Foong et al., 2019), which have shown that some UQ methods might not yield high quality uncertainty estimates due to the learning paradigm or sensitivity to parameters. Let us look at each column more carefully:

1. DE tends to identify the OOD examples as the most uncertain examples. Specifically, looking at the bottom figure we see that the top 5% untrusted examples identified by DE are almost all OOD examples, which is not the case for the other UQ methods. By contrast, MCD is poor at identifying OOD examples; it flags some of the OOD samples as the most certain. This is explained by MCD's mechanism. Uncertainty is based on the difference in predictions across different drop-outs, however this could lead to outliers always having the same prediction— due to correlation between different nodes in the MCD model, extreme values of the OOD examples may always saturate the network and lead to the same prediction, even if some nodes are dropped out. DE is most apt at flagging the OOD class. This confirms the finding by (Ovadia et al., 2019) who showed that DE outperforms other methods under dataset shift—which is effectively what OOD represents.

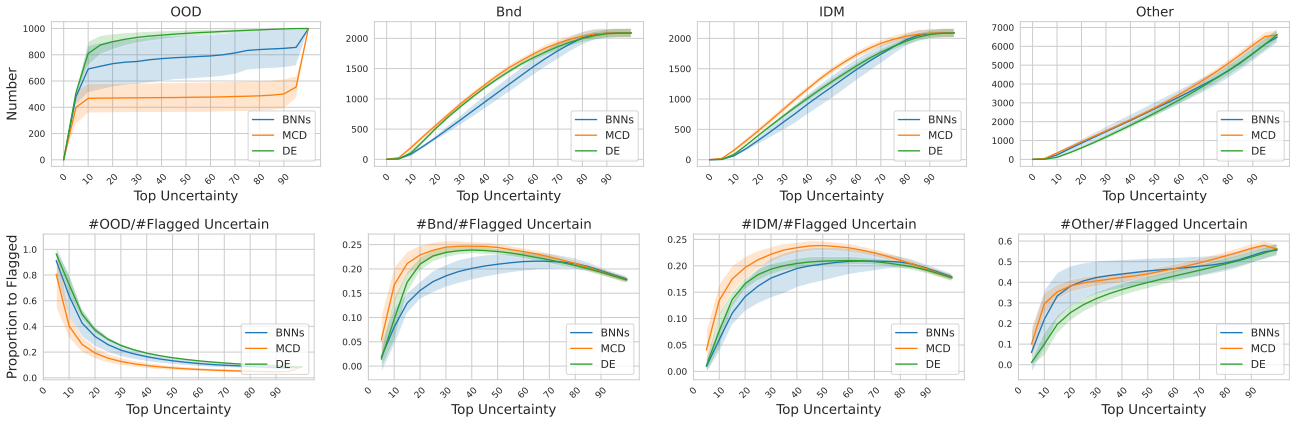


Figure 6. Results of applying our method in explaining different UQ methods. First row: comparisons on the numbers in different classes of examples. Second row: comparisons on the proportion of different classes of flagged examples to the total number of identified uncertain examples. Different methods tend to identify different certain type of uncertain examples.

2. After the OOD examples have been flagged, the next most suspicious examples are the Bnd and IDM classes—see columns 2 and 3. The number of these examples increases almost linearly with the number of flagged examples, until at about 88% no more examples are flagged as IDM and Bnd. This behaviour is explained by the Vanilla MNIST examples—which are generally distinguishable and relatively easily classified correctly—accounting for about 15% of the test examples.

3. As expected, the number of examples belonging to the *Other* class increases when more examples are flagged as uncertain. This makes sense, as the *Other* class indicates we cannot explain why the methods flagged these examples as uncertain, i.e. maybe these predictions should in fact be trusted.

5.4. Improving Uncertain Predictions

In this section, we explore the inverse direction, i.e. employing DAUX's explanations for creating better models. We use UCI's **Covtype** and **Digits** datasets (Dua & Graff, 2017) and focus on linear models (i.e. $g = Id$), since linear models are widely applied in settings where uncertainty, risk and misclassification are crucial—e.g. healthcare and finance. Moreover, a linear model allows us to disentangle the benefit of DAUX's proposed inverse method from the possible random performance differences based on different representations. We use DE as the baseline uncertainty model. In Appendix B we show (Figure 8) that the B&I examples are generally hardest to classify, hence let us focus on improving the predictive performance on this class.

We vary the proportion q of training samples that we discard before training new prediction model $f_{\text{B\&I}}$—only saving the training samples that resemble the B&I dataset most—see Figure 7. We find that retraining the linear model with filtered training data according to Eq. 6 significantly improves the performance. We observe that performance increases approximately linearly proportional to q, until the amount of training data becomes too low. The latter depends on the dataset and model used. See Appendix B.2 for more details.

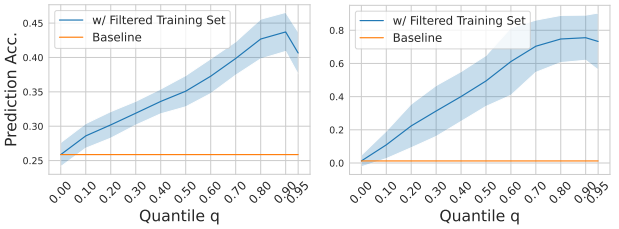


Figure 7. Improved uncertain predictions. Left: **Covtype**, Right: **Digits**. Experiment are performed with different proportion of training samples discarded: e.g., with $q = 0.0$, all examples in the training set are used; while with $q = 0.9$, only top 10% examples most resembling the test data are used for training.

6. Conclusion

We have proposed DAUX, a framework for explaining model uncertainty. DAUX categorizes uncertain examples identified by UQ benchmarks into three classes—OOD, Bnd and IDM. These classes correspond to different causes for the uncertainty and require different strategies for possibly better predictions. We have demonstrated the power of DAUX by explaining three different UQ benchmarks—highlighting that each one identifies different examples—as well as by using DAUX's explanations to improve prediction performance on multiple datasets. We believe DAUX can aid the development of trustworthy models and we hope it paves the way for further advances in explaining uncertainty quantification methods.

References

Abdar, M., Pourpanah, F., Hussain, S., Rezazadegan, D., Liu, L., Ghavamzadeh, M., Fieguth, P., Cao, X., Khosravi, A., Acharya, U. R., Makarenkov, V., and Nahavandi, S. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 76:243–297, 2021a. ISSN 1566-2535. doi: 10.1016/j.inffus.2021.05.008.

Abdar, M., Pourpanah, F., Hussain, S., Rezazadegan, D., Liu, L., Ghavamzadeh, M., Fieguth, P., Cao, X., Khosravi, A., Acharya, U. R., et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 2021b.

Aggarwal, R., Sounderajah, V., Martin, G., Ting, D. S., Karthikesalingam, A., King, D., Ashrafian, H., and Darzi, A. Diagnostic accuracy of deep learning in medical imaging: a systematic review and meta-analysis. *NPJ digital medicine*, 4(1):1–23, 2021.

Balasubramanian, V., Ho, S.-S., and Vovk, V. *Conformal prediction for reliable machine learning: theory, adaptations and applications*. Newnes, 2014.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural network. In *International Conference on Machine Learning*, pp. 1613–1622. PMLR, 2015.

Crabbé, J., Qian, Z., Imrie, F., and van der Schaar, M. Explaining latent representations with a corpus of examples. *NeurIPS 2021*, 2021.

Deng, W. and Zheng, L. Are labels always necessary for classifier accuracy evaluation? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15069–15078, 2021.

Ding, Q., Wu, S., Sun, H., Guo, J., and Guo, J. Hierarchical multi-scale gaussian transformer for stock movement prediction. In *IJCAI*, pp. 4640–4646, 2020.

Dua, D. and Graff, C. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Ekambaram, R., Goldgof, D. B., and Hall, L. O. Finding label noise examples in large scale datasets. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 2420–2424. IEEE, 2017.

Foong, A. Y., Burt, D. R., Li, Y., and Turner, R. E. On the expressiveness of approximate inference in bayesian neural networks. *arXiv preprint arXiv:1909.00719*, 2019.

Gal, Y. and Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059. PMLR, 2016.

Gao, B. and Pavel, L. On the properties of the softmax function with application in game theory and reinforcement learning. *arXiv preprint arXiv:1704.00805*, 2017.

Ghanta, S., Subramanian, S., Khermosh, L., Shah, H., Goldberg, Y., Sundararaman, S., Roselli, D., and Talagala, N. {MPP}: Model performance predictor. In *2019 {USENIX} Conference on Operational Machine Learning (OpML 19)*, pp. 23–25, 2019.

Ghosh, S., Yao, J., and Doshi-Velez, F. Structured variational learning of bayesian neural networks with horseshoe priors. In *International Conference on Machine Learning*, pp. 1744–1753. PMLR, 2018.

Ghosh, S., Liao, Q. V., Ramamurthy, K. N., Navratil, J., Sattigeri, P., Varshney, K. R., and Zhang, Y. Uncertainty quantification 360: A holistic toolkit for quantifying and communicating the uncertainty of ai, 2021.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Graves, A. Practical variational inference for neural networks. *Advances in neural information processing systems*, 24, 2011.

Hüllermeier, E. and Waegeman, W. Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods. *Machine Learning 2021 110:3*, 110(3):457–506, 2021. ISSN 1573-0565. doi: 10.1007/S10994-021-05946-3.

Kendall, A. and Gal, Y. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in Neural Information Processing Systems*, 30:5574–5584, 2017.

Kim, A., Yang, Y., Lessmann, S., Ma, T., Sung, M.-C., and Johnson, J. E. Can deep learning predict risky retail investors? a case study in financial risk behavior forecasting. *European Journal of Operational Research*, 283(1): 217–234, 2020.

Kim, J. and Canny, J. Interpretable learning for self-driving cars by visualizing causal attention. In *Proceedings of the IEEE international conference on computer vision*, pp. 2942–2950, 2017.

Kishida, I. and Nakayama, H. Empirical study of easy and hard examples in cnn training. In *International Conference on Neural Information Processing*, pp. 179–188. Springer, 2019.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Proceedings of the 31st International*

Conference on Neural Information Processing Systems, pp. 6405–6416, 2017.

Lee, K., Lee, K., Lee, H., and Shin, J. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. *Advances in neural information processing systems*, 31, 2018.

Liu, F. T., Ting, K. M., and Zhou, Z.-H. Isolation forest. In *2008 eighth ieee international conference on data mining*, pp. 413–422. IEEE, 2008.

Malinin, A. and Gales, M. Predictive uncertainty estimation via prior networks. *arXiv preprint arXiv:1802.10501*, 2018.

Mukhoti, J., Kirsch, A., van Amersfoort, J., Torr, P. H., and Gal, Y. Deterministic neural networks with appropriate inductive biases capture epistemic and aleatoric uncertainty. *arXiv preprint arXiv:2102.11582*, 2021.

Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J., Lakshminarayanan, B., and Snoek, J. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *Advances in Neural Information Processing Systems*, 32:13991–14002, 2019.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

Ramalho, T. and Miranda, M. Density estimation in representation space to predict model uncertainty. In *International Workshop on Engineering Dependable and Secure Machine Learning Systems*, pp. 84–96. Springer, 2020.

Ran, X., Xu, M., Mei, L., Xu, Q., and Liu, Q. Detecting out-of-distribution samples via variational auto-encoder with reliable uncertainty estimation. *Neural Networks*, 145:199–208, 2022.

Rasmussen, C. E. Gaussian processes in machine learning. In *Summer school on machine learning*, pp. 63–71. Springer, 2003.

Ren, J., Liu, P. J., Fertig, E., Snoek, J., Poplin, R., Depristo, M., Dillon, J., and Lakshminarayanan, B. Likelihood ratios for out-of-distribution detection. *Advances in Neural Information Processing Systems*, 32:14707–14718, 2019.

Schut, L., Key, O., Mc Grath, R., Costabello, L., Sacaleanu, B., Gal, Y., et al. Generating interpretable counterfactual explanations by implicit minimisation of epistemic and aleatoric uncertainties. In *International Conference on Artificial Intelligence and Statistics*, pp. 1756–1764. PMLR, 2021.

Scott, D. W. Multivariate density estimation and visualization. In *Handbook of computational statistics*, pp. 549–569. Springer, 2012.

Sheather, S. J. and Jones, M. C. A reliable data-based bandwidth selection method for kernel density estimation. *Journal of the Royal Statistical Society: Series B (Methodological)*, 53(3):683–690, 1991.

Silverman, B. W. *Density estimation for statistics and data analysis*. Routledge, 2018.

Sun, J., Sun, H., Han, T., and Zhou, B. Neuro-symbolic program search for autonomous driving decision module design. In Kober, J., Ramos, F., and Tomlin, C. (eds.), *Proceedings of the 2020 Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pp. 21–30. PMLR, 16–18 Nov 2021. URL https://proceedings.mlr.press/v155/sun21a.html.

Tsipras, D., Santurkar, S., Engstrom, L., Ilyas, A., and Madry, A. From imagenet to image classification: Contextualizing progress on benchmarks. In *International Conference on Machine Learning*, pp. 9625–9635. PMLR, 2020.

Van Amersfoort, J., Smith, L., Teh, Y. W., and Gal, Y. Uncertainty estimation using a single deep deterministic neural network. In *International Conference on Machine Learning*, pp. 9690–9700. PMLR, 2020.

Van Looveren, A., Klaise, J., Vacanti, G., Cobb, O., Scillitoe, A., and Samoilescu, R. Alibi detect: Algorithms for outlier, adversarial and drift detection, 2019. URL https://github.com/SeldonIO/alibi-detect.

Verdoja, F. and Kyrki, V. Notes on the behavior of mc dropout. *arXiv preprint arXiv:2008.02627*, 2020.

Vovk, V., Gammerman, A., and Shafer, G. *Algorithmic learning in a random world*. Springer Science & Business Media, 2005.

Yao, J., Pan, W., Ghosh, S., and Doshi-Velez, F. Quality of uncertainty quantification for bayesian neural network inference. *arXiv preprint arXiv:1906.09686*, 2019.

A. Missing Details

A.1. Motivations for working with model latent space

In Section 3, we introduced the confusion density matrix that allows us to categorize suspicious examples at testing time. Crucially, this density matrix relies on kernel density estimations in the latent space $\mathcal{H}$ associated to the model f through Assumption 1. Why are we performing a kernel density estimation in latent space rather than in input space $\mathcal{X}$? The answer is fairly straightforward: we want our density estimation to be coupled to the model and its predictions.

Let us now make this point more rigorous. Consider two input examples $x_1, x_2 \in \mathcal{X}$. The model assigns a representations $g(x_1), g(x_2) \in \mathcal{H}$ and class probabilities $f(x_1), f(x_2) \in \mathcal{Y}$. If we define our kernel κ in latent space $\mathcal{H}$, this often means[2] that $\kappa[g(x_1), g(x_2)]$ grows as $\|g(x_1) - g(x_2)\|_{\mathcal{H}}$ decreases. Hence, examples that are assigned a similar latent representation by the model f are related by the kernel. Since our whole discussion revolves around model *predictions*, we would like to guarantee that two examples related by the kernel are given similar predictions by the model f. In this way, we would be able to interpret a large kernel density $\kappa[g(x_1), g(x_2)]$ as a hint that the predictions $f(x_1)$ and $f(x_2)$ are similar. We will now show that, under Assumption 1, such a guarantee exists. Similar to (Crabbé et al., 2021), we start by noting that

$$\begin{aligned}\|(l \circ g)(x_1) - (l \circ g)(x_2)\|_{\mathbb{R}^C} &= \|l\,[g(x_1) - g(x_2)]\|_{\mathbb{R}^C} \\ &\leq \|l\|_{\text{op}} \|g(x_1) - g(x_2)\|_{\mathcal{H}}\,,\end{aligned}$$

where $\|\cdot\|_{\mathbb{R}^C}$ is a norm on $\mathbb{R}^C$ and $\|l\|_{\text{op}}$ is the operator norm of the linear map l. In order to extend this inequality to black-box predictions, we note that the normalizing map in Assumption 1 is often a Lipschitz function with Lipschitz constant $\lambda \in \mathbb{R}$. For instance, a Softmax function with inverse temperature constant λ^{-1} is λ-Lipschitz (Gao & Pavel, 2017). We use this fact to extend our inequality to predicted class probabilities:

$$\begin{aligned}\|f(x_1) - f(x_2)\|_{\mathcal{Y}} &= \|(\varphi \circ l \circ g)(x_1) - (\varphi \circ l \circ g)(x_2)\|_{\mathcal{Y}} \\ &\leq \lambda\, \|(l \circ g)(x_1) - (l \circ g)(x_2)\|_{\mathbb{R}^C} \\ &\leq \lambda\, \|l\|_{\text{op}} \|g(x_1) - g(x_2)\|_{\mathcal{H}}\,.\end{aligned}$$

This crucial inequality guarantees that examples $x_1, x_2 \in \mathcal{X}$ that are given a similar latent representation $g(x_1) \approx g(x_2)$ will also be given a similar prediction $f(x_1) \approx f(x_2)$. In short: two examples that are related according to a kernel density defined in the model latent space $\mathcal{H}$ are guaranteed to have similar predictions. This is the motivation we wanted to support the definition of the kernel κ in latent space.

An interesting question remains: is it possible to have similar guarantees if we define the kernel in input space? When we deal with deep models, the existence of adversarial examples indicates the opposite (Goodfellow et al., 2014). Indeed, if x_2 is an adversarial example with respect to x_1, we have $x_1 \approx x_2$ (and hence $\|x_1 - x_2\|_{\mathcal{X}}$ small) with two predictions $f(x_1)$ and $f(x_2)$ that are significantly different. Therefore, defining the kernel κ in input space might result in relating examples that are given a significantly different prediction by the model. For this reason, we believe that the latent space is more appropriate in our setting.

A.2. Details: Flagging IDM and Bnd Examples with Thresholds

In order to understand uncertainty, it will be clearer to map those scores into binary classes with thresholds. In our experiments, we use empirical quantiles as thresholds. e.g., to label an example as IDM, we specify an empirical quantile number q, and calculate the corresponding threshold based on the order statistics of IDM Scores for test examples: $S_{\text{IDM}}^{(1)}, ..., S_{\text{IDM}}^{(|\mathcal{D}_{\text{test}}|)}$, where $S_{\text{IDM}}^{(n)}$ denotes the n-th smallest IDM score out of $|\mathcal{D}_{\text{test}}|$ testing-time examples. Then, the threshold given quantile number q is

$$\tau_{\text{IDM}}(q) \equiv S_{\text{IDM}}^{(\lfloor |\mathcal{D}_{\text{test}}| \cdot q \rfloor)}.$$

Similarly, we can define quantile-based threshold in flagging Bnd examples based on the order statistics of Bnd Scores for test examples, such that for given quantile q,

$$\tau_{\text{Bnd}}(q) \equiv S_{\text{Bnd}}^{(\lfloor |\mathcal{D}_{\text{test}}| \cdot q \rfloor)}.$$

[2] This is the case for all the kernels that rely on a distance (e.g. the Radial Basis Function Kernel, the Matern kernel or even Polynomial kernels (Rasmussen, 2003)).

Practically, a natural choice of q is to use the validation accuracy: when there are $1-q$ examples misclassified in the validation set, we also expect the testing-time in distribution examples with the highest $1-q$ to be marked as Bnd or IDM examples.

A.3. Inverse Direction

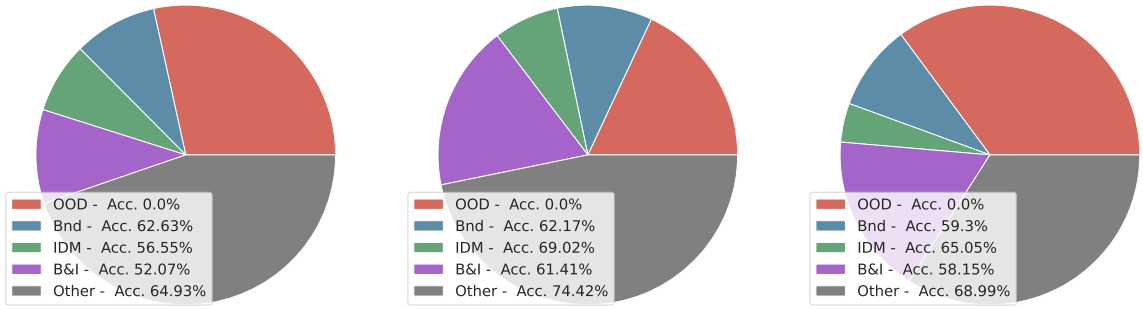
Threshold $\tau_{\text{test}}(q)$ For every training example $x \in \mathcal{D}_{\text{train}}$, we have the latent density $p(x|\mathcal{D}_{\text{B\&I}})$ over the B&I class of the test set. With their order statistics $p_{(1)}(x|\mathcal{D}_{\text{B\&I}}), ..., p_{(|\mathcal{D}_{\text{train}}|)}(x|\mathcal{D}_{\text{B\&I}})$. Given quantile number q, our empirical quantile based threshold τ_{test} is chosen as

$$\tau_{\text{test}}(q) \equiv p_{(\lfloor q \cdot |\mathcal{D}_{\text{train}}| \rfloor)}(x|\mathcal{D}_{\text{B\&I}}).$$

During the inverse training time, we train our model to predict those B&I class of test examples only with the training data with higher density than $\tau_{\text{test}}(q)$. We experiment with different choices of q in the experiment (Figure 7 in Sec. 5.4).

B. Additional Experiments

B.1. Explanations of Uncertainty under Different Thresholds



(a) BNNs (b) MCD (c) DE

Figure 8. The top 25% uncertain examples identified by different methods. Legend of each figure provide the accuracy and proportion of each class. As the classifier can not make correct predictions on the OOD examples, it's always better for uncertainty estimators to flag more OOD examples.

In the main text, we provide results with $\tau_{\text{Bnd}} = \tau_{\text{IDM}} = 0.8$, which approximates the accuracy on validation set—as a natural choice. In this section, we vary these thresholds and show that changing those thresholds does not significantly alter the conclusions drawn above.

Figure 8 looks more closely into the top 25% uncertain examples for each method, and the accuracy on each of the uncertainty classes. As expected, the accuracy of the B&I examples is always lower than that of the trusted class, meaning that those examples are most challenging for the classifier. And the accuracy of flagged classes are always lower than the *other* class, verifying the proposed explanation of different classes.

B.2. Inverse Direction: More Results

In the main text, we show the results on improving prediction performance on the B&I class with training example filtering (On the Covtype, Digits dataset). More results on other classes of examples are provided in this section.

We experiment on three UCI datasets: **Covtype**, **Digits**, and **Spam**. And experiment with three classes we defined in this work:

1. **B&I** class (Figure 10). As we have discussed in our main text, the prediction accuracy on the B&I class are always the lowest among all classes. By training with filtered examples in $\mathcal{D}_{\text{train}}$ rather than the entire training set, the B&I class of examples can be classified with a remarkably improved accuracy.

Table 4. DAUX is not the only choice in identifying OOD examples. On the Dirty-MNIST dataset, DAUX, Outlier-AE and the IForest can identify most outliers in the test dataset. (Given threshold = 1.0 for those two benchmark methods).

Method	Precision	Recall	F1-Score
DAUX	1.0000 ± 0.0000	1.0000 ± 0.0000	1.0000 ± 0.0000
Outlier-AE	1.0000 ± 0.0000	1.0000 ± 0.0000	1.0000 ± 0.0000
IForest (Liu et al., 2008)	0.9998 ± 0.0004	1.0000 ± 0.0000	0.9999 ± 0.0002

2. **Bnd** class (Figure 11). This class of examples are located at boundaries in the latent space of validation set, but not necessarily have been misclassified. Therefore, their performance baseline (training with the entire $\mathcal{D}_{\text{train}}$) is relatively high. The improvement is clear but not as much as on the other two classes.

3. **IDM** class (Figure 12). For this class of examples, similar mistakes have been make in the validation set, yet those examples are not necessarily located in the boundaries—the misclassification may be caused by ambiguity in decision boundary, imperfectness of either the model or the dataset. The primal prediction accuracy on this class of examples is lower than the Bnd class but higher than the B&I class, training with filtered $\mathcal{D}_{\text{train}}$ also clearly improve the performance on this class of examples.

B.3. Alternative Approach in Flagging OOD

As we have mentioned in the main text, although DAUX has a unified framework in understanding all three types of uncertainty the uncertain caused by OOD examples can also be identified by off-the-shelf algorithms. We compare DAUX to two existing outlier detection methods in Table 4, where all methods achieve good performance on the Dirty-MNIST dataset. Our implementation is based on Alibi Detect (Van Looveren et al., 2019).

C. Implementation Details

C.1. Hyperparameters

C.1.1. Bandwidth

In our experiments, we use (z-score) normalized latent representations and bandwidth 1.0. In the inverse direction, as the sample sizes are much smaller, a bandwidth of 0.01 is used as the recommended setting. There is a vast body of research on selecting a good bandwidth for Kernel Density Estimation models (Sheather & Jones, 1991; Scott, 2012; Silverman, 2018) and using these to adjust DAUX's bandwidth to a more informed choice may further improve performance.

C.2. Inverse Direction: Quantile Threshold q

As depicted in Appendix A.2, a natural choice of q is to use the validation accuracy. We use this heuristic approach in our experiments for the inverse direction.

C.3. Model Structure

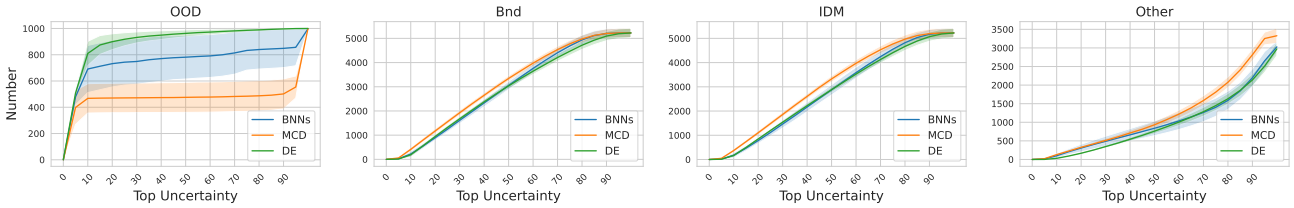
In our experiments, we implement **MCD** and **DE** with 3-layer-CNNs with ReLU activation. Our experiments on **BNNs** are based on the IBM UQ360 software (Ghosh et al., 2021). More details of the convolutional network structure are provided in Table 5.

C.4. Implementation of Kernel Density Estimation and Repeat Runs

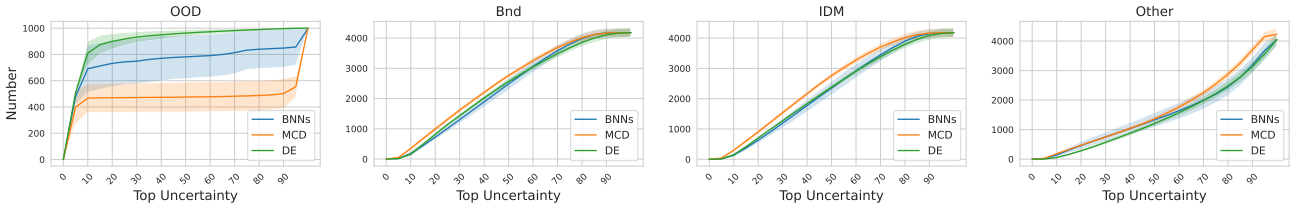
Our implementation of KDE models are based on the sklearn's KDE package (Pedregosa et al., 2011). Gaussian kernels are used as default settings. In all experiments, we run with 10 random seeds and report the averaged results.

Table 5. Network Structure

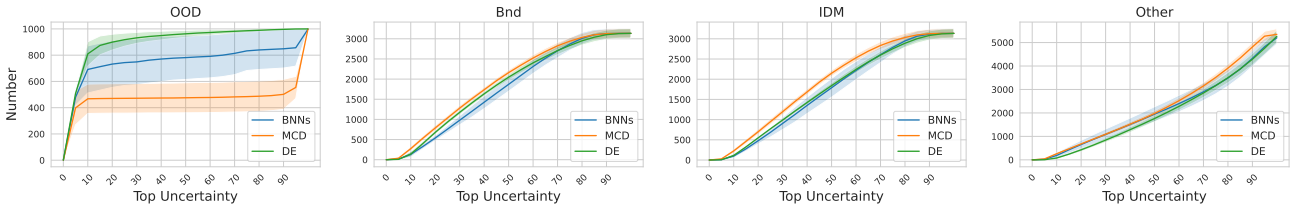
Layer	Unit	Activation	Pooling
Conv 1	$(1, 32, 3, 1, 1)$	ReLU()	MaxPool2d(2)
Conv 2	$(32, 64, 3, 1, 1)$	ReLU()	MaxPool2d(2)
Conv 3	$(64, 64, 3, 1, 1)$	ReLU()	MaxPool2d(2)
FC	$(64 \times 3 \times 3, 40)$	ReLU()	-
Out	$(40, N_{\text{Class}})$	SoftMax()	-



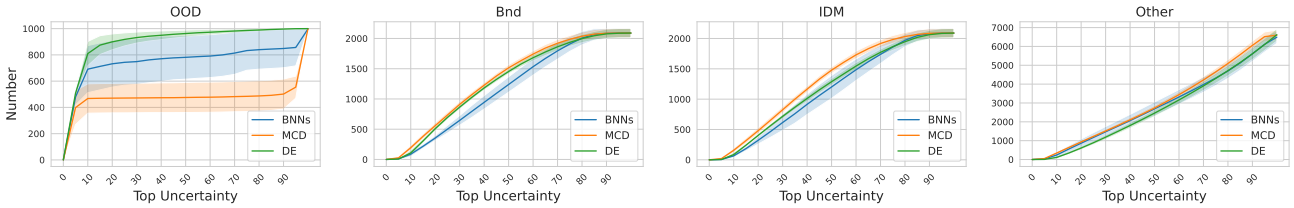
(a) $\tau_{\text{BD}}(0.5), \tau_{\text{IDM}}(0.5)$



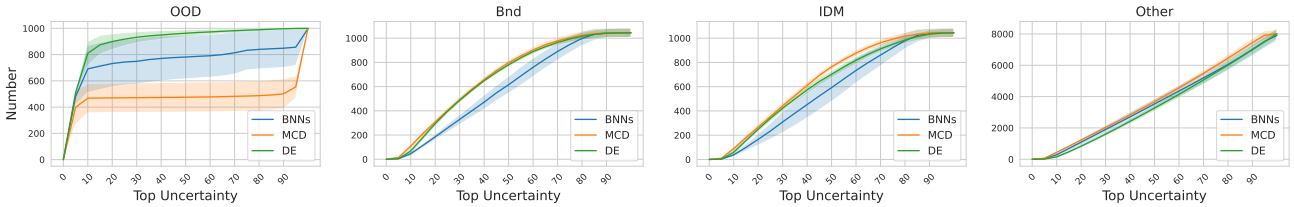
(b) $\tau_{\text{BD}}(0.6), \tau_{\text{IDM}}(0.6)$



(c) $\tau_{\text{BD}}(0.7), \tau_{\text{IDM}}(0.7)$

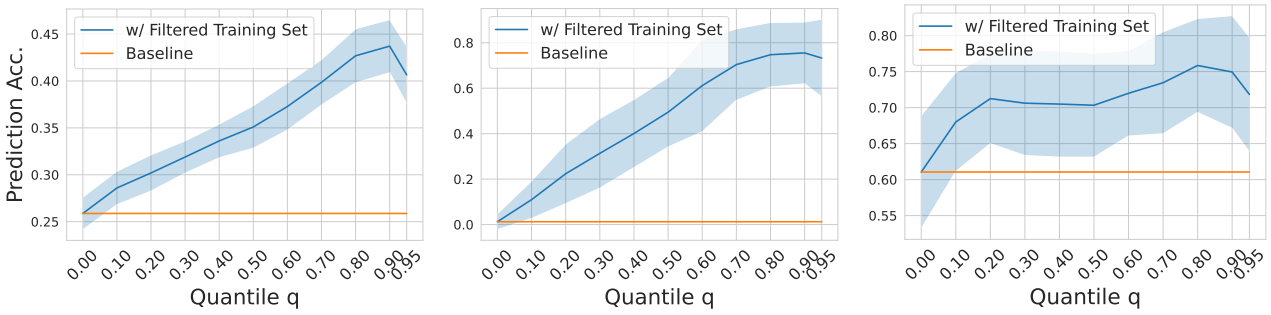


(d) $\tau_{\text{BD}}(0.8), \tau_{\text{IDM}}(0.8)$



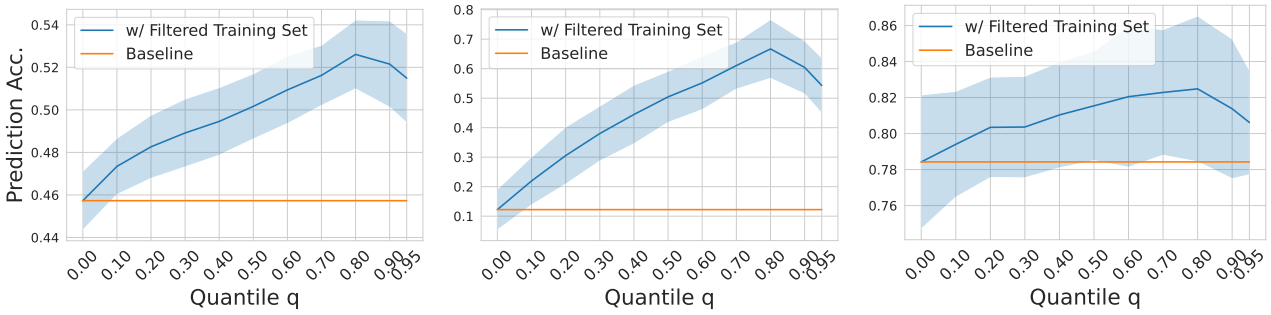
(e) $\tau_{\text{BD}}(0.9), \tau_{\text{IDM}}(0.9)$

Figure 9. Experiments on different choices of thresholds.



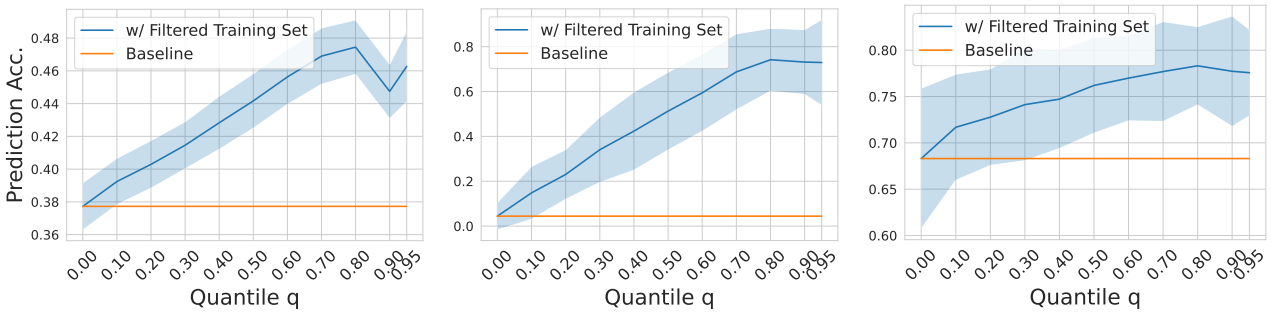
(a) Covtype (b) Digits (c) Spam

Figure 10. Experiments on the B&I class (reported in the main text).



(a) Covtype (b) Digits (c) Spam

Figure 11. Experiments on the Bnd class.



(a) Covtype (b) Digits (c) Spam

Figure 12. Experiments on the IDM class.