Creating and Evaluating Uncertainty Estimates with Neural Networks for Environmental-Science Applications

KATHERINE HAYNES,[a] RYAN LAGERQUIST,[a,b] MARIE MCGRAW,[a] KATE MUSGRAVE,[a] AND IMME EBERT-UPHOFF[a,c]

[a] *Cooperative Institute for Research in the Atmosphere, Colorado State University, Fort Collins, Colorado*
[b] *NOAA/Earth System Research Laboratory/Global Systems Laboratory, Boulder, Colorado*
[c] *Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, Colorado*

(Manuscript received 9 August 2022, in final form 4 January 2023)

ABSTRACT: Neural networks (NN) have become an important tool for prediction tasks—both regression and classification—in environmental science. Since many environmental-science problems involve life-or-death decisions and policy making, it is crucial to provide not only predictions but also an estimate of the uncertainty in the predictions. Until recently, very few tools were available to provide uncertainty quantification (UQ) for NN predictions. However, in recent years the computer-science field has developed numerous UQ approaches, and several research groups are exploring how to apply these approaches in environmental science. We provide an accessible introduction to six of these UQ approaches, then focus on tools for the next step, namely, to answer the question: *Once we obtain an uncertainty estimate (using any approach), how do we know whether it is good or bad?* To answer this question, we highlight four evaluation graphics and eight evaluation scores that are well suited for evaluating and comparing uncertainty estimates (NN based or otherwise) for environmental-science applications. We demonstrate the UQ approaches and UQ-evaluation methods for two real-world problems: 1) estimating vertical profiles of atmospheric dewpoint (a regression task) and 2) predicting convection over Taiwan based on *Himawari-8* satellite imagery (a classification task). We also provide Jupyter notebooks with Python code for implementing the UQ approaches and UQ-evaluation methods discussed herein. This article provides the environmental-science community with the knowledge and tools to start incorporating the large number of emerging UQ methods into their research.

SIGNIFICANCE STATEMENT: Neural networks are used for many environmental-science applications, some involving life-or-death decision-making. In recent years new methods have been developed to provide much-needed uncertainty estimates for NN predictions. We seek to accelerate the adoption of these methods in the environmental-science community with an accessible introduction to 1) methods for computing uncertainty estimates in NN predictions and 2) methods for evaluating such estimates.

KEYWORDS: Uncertainty; Artificial intelligence; Data science; Deep learning; Neural networks

1. Introduction

Neural networks (NN), a type of machine learning (ML) model, have become widely used in environmental science, including both regression and classification tasks. Many such tasks—predicting ocean-wave heights, rapid intensification of hurricanes, formation of tornadoes, etc.—are crucial for decision-making and policy making. In order for people to make such important decisions, ML models need to provide not only the predicted outcome, but also the uncertainty in the prediction.

Because accurate and reliable weather forecasts are important to a wide range of communities, the meteorology community long ago recognized the importance of probabilistic predictions. Numerical weather prediction (NWP) models are often run in ensemble mode to produce multiple forecasts; averaging over the ensemble often improves forecast quality, but the ensemble can also be used to quantify forecast uncertainty. To calibrate the forecast uncertainty—for example, to correct biases in the ensemble spread—many statistical post-processing techniques have been developed (see Vannitsem et al. 2021 for an overview). Most of these techniques are agnostic to the underlying models and can therefore be trivially adapted to NN models rather than NWP models.

Until recently, very few tools existed to provide uncertainty quantification (UQ) for NN predictions. However, in recent years the computer-science field has made rapid advancements in this area. The current article aims to help the environmental-science community apply UQ techniques for NNs and relate them to preexisting techniques for postprocessing NWP ensembles.

To achieve this, we first provide a background on what types of uncertainty can be estimated by NNs for environmental-science applications (section 2). Next, we provide an accessible introduction to six UQ approaches (section 3): parametric distributional prediction (PDP), nonparametric distributional prediction (NPDP), ensemble prediction (EP), multimodel (MM),

Supplemental information related to this paper is available at the Journals Online website: https://doi.org/10.1175/AIES-D-22-0061.s1.

Corresponding author: Katherine Haynes, katherine.haynes@colostate.edu.

DOI: 10.1175/AIES-D-22-0061.1 e220061

Monte Carlo (MC) dropout, and Bayesian neural networks (BNN). In section 4 we discuss tools for the next step, namely answering the question: *Once we obtain an uncertainty estimate (using any approach), how we do evaluate the quality of this estimate?* We highlight four evaluation graphics and eight evaluation scores that are well suited for environmental-science applications:

- The attributes diagram, including the mean-squared-error (MSE) skill score (MSESS);
- the spread–skill plot, including the spread–skill ratio (SSRAT) and spread–skill reliability (SSREL);
- the discard test, including the monotonicity fraction (MF) and average discard improvement (DI);
- the probability integral transform (PIT) histogram, including the calibration deviation (PITD);
- the continuous ranked probability score (CRPS); and
- the ignorance score (IGN).

In sections 5 and 6 we demonstrate the UQ approaches and UQ-evaluation methods for two real-world problems: estimating vertical profiles of atmospheric dewpoint (a regression task) and predicting convection over Taiwan based on satellite imagery (a classification task). The first application builds UQ into the work of Stock (2021), while the second builds UQ into the work of Lagerquist et al. (2021). Finally, in section 7, we provide insights gained throughout this work on both the UQ approaches and UQ-evaluation methods.

This article is accompanied by four Jupyter notebooks for implementing both the UQ approaches and UQ-evaluation methods discussed (https://github.com/thunderhoser/cira_uq4ml). For classification tasks, the MC dropout notebook (classification_mc_dropout.ipynb) implements the MC-dropout method, and the NPDP notebook (classification_npdp.ipynb) implements quantile regression with a special NN architecture that prevents quantile crossing. Both notebooks also implement the spread–skill plot and discard test. For regression tasks, there are two notebooks, both using six synthetic datasets and three UQ approaches (PDP, EP, and MC dropout). The first (regression_multi_datasets.ipynb) allows the user to select one UQ approach, then compares the results across the datasets, using all evaluation tools (graphics and scores) listed above. The second (regression_multi_model.ipynb) allows the user to select a dataset, then compares the results across three UQ approaches using all evaluation tools.

2. Background

a. *Which uncertainties are we trying to capture?*

The motivation for quantifying uncertainty in ML is to provide information on how much to trust the model's prediction. To identify what *types* of uncertainty we can expect an ML model to provide, first we must look at the model structure, sources of uncertainty, and pieces of information the model can use to quantify uncertainty (Fig. 1). In Fig. 1 the training data consist of pairs ($\mathbf{x}$, y_{true}); $\mathbf{x}$ is the input vector, containing predictors (or "features"), and y_{true} is the desired output (or "target value" or "label" or "ground truth"). During training, the ML model learns to generate better and better predictions by minimizing a loss function, which measures the error between the target values y_{true} and predicted values y_{pred}. At inference time, a deterministic ML model produces one prediction per data sample $\mathbf{x}$. This approach works in an idealized setting, where 1) the model can perfectly learn the conditional distribution $y_{\text{true}}|\mathbf{x}$ during training; 2) this relationship does not change between the training and inference data; and 3) the predictor distribution $\mathbf{x}$ does not change between the training and inference data.

However, the idealized setting rarely occurs, for several reasons. First, datasets include numerous sources of uncertainty, leading to many y_{true} possibilities for each predictor vector $\mathbf{x}$; second, different model structures may compromise model performance; third, the predictor distribution at inference time may contain out-of-regime samples, forcing the model to extrapolate outside the range of the training data; fourth, the relationship $y_{\text{true}}|\mathbf{x}$ might change between the training and inference data.

Ideally, a model's uncertainty estimates would account for contributions from all four error sources. However, only two pieces of information are available to derive uncertainty estimates: the data and the ML model. Thus, we task the ML model with providing an uncertainty estimate that accounts for two error sources (Fig. 2). The first is internal variability in the data, resulting from stochasticity in physical processes. This manifests as spread in the target value y_{true} for a given predictor vector $\mathbf{x}$ (middle purple container in Fig. 1; left side of Fig. 2). The model can learn this spread and quantify it for each data sample. The second source of uncertainty is out-of-regime error; the model should produce higher uncertainty estimates for data samples $\mathbf{x}$ outside the range of the training data (rightmost purple container in Fig. 1; right side of Fig. 2).

Although we want the ML model to simultaneously quantify uncertainties stemming from all error sources, we emphasize that the first step is to reduce these errors as much as possible. This can be done by 1) collecting data from all relevant regimes, to minimize the occurrence of out-of-regime data samples at inference time; 2) carefully processing the data, including quality control; and 3) optimizing the model structure to achieve the best performance possible.

b. *Aleatory versus epistemic uncertainty*

In environmental science, uncertainty sources are often divided into four categories (Beucler et al. 2021): 1) stochastic uncertainty, due to internal variability such as randomness arising from the chaotic nature of fluid flows; 2) observational uncertainty, due to measurement and representation errors; 3) structural uncertainty, due to incorrect model structure; and 4) parametric uncertainty, due to incorrect model parameters.

Other classifications distinguish only two types of uncertainty—aleatory and epistemic—but the line dividing the two types differs by discipline. Figure 3 illustrates this difference for two disciplines: mathematics versus ML. In the original math context, aleatory and epistemic uncertainty are defined as follows:

- *Aleatory* comes from the Latin word *alea*, which refers to a game of dice. *Aleatory uncertainty in mathematics* refers to the

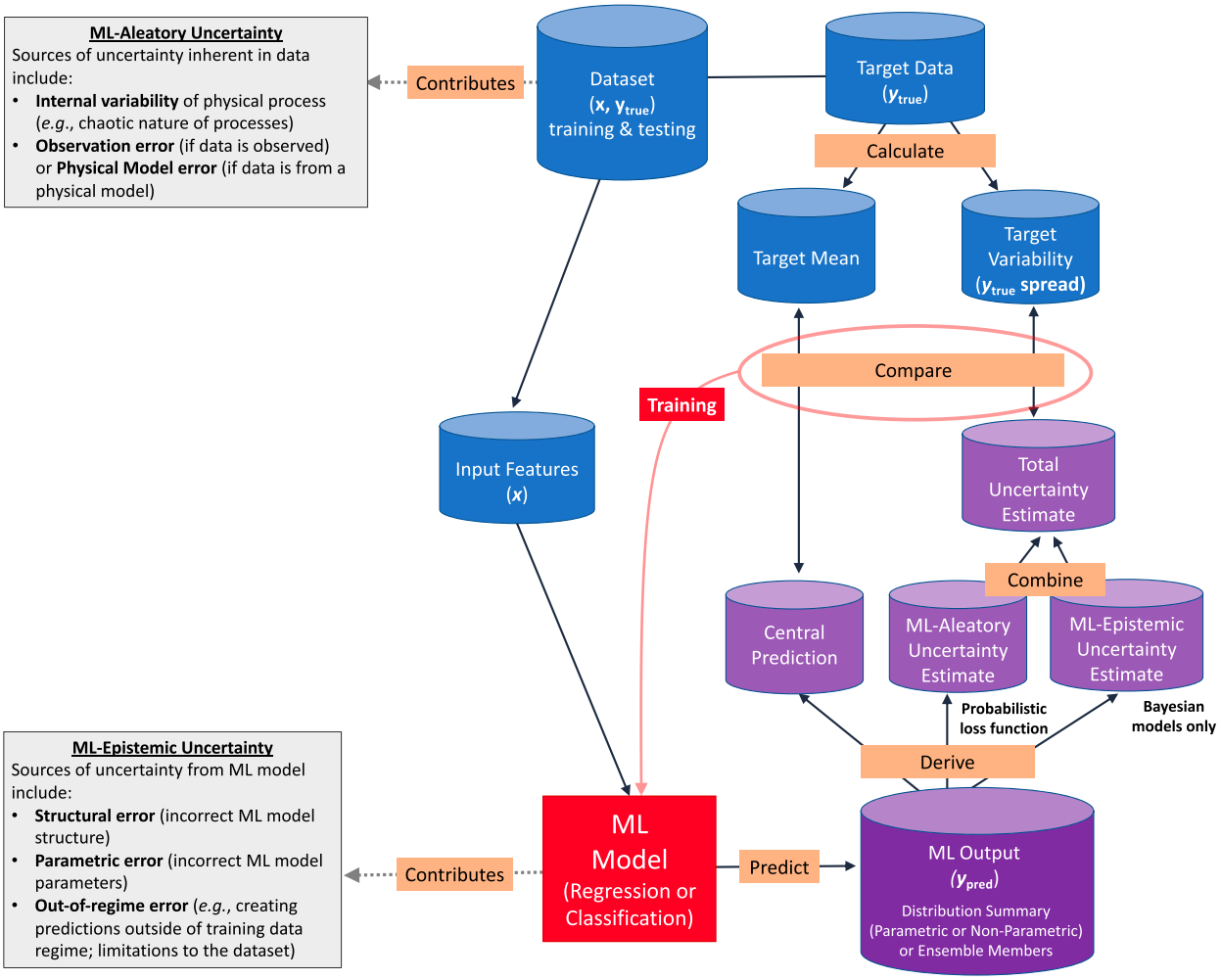


FIG. 1. Information flow for an ML model with UQ (i.e., one that provides both a central prediction and uncertainty information). The uncertainty information may take one of three forms: an ensemble of predictions (considered samples from the y_{pred} distribution), a parametric summary of the y_{pred} distribution (e.g., the mean and standard deviation, assuming a normal distribution), or a nonparametric summary of the y_{pred} distribution (e.g., a set of quantiles). Blue containers represent data, the red box represents the ML model, and purple containers represent model output. To indicate how much users should trust the ML model's prediction, UQ approaches seek to estimate the total uncertainty present for each data sample.

stochastic component of uncertainty that stems from randomness in the data-generation process, such as the chaotic nature of fluid flows in the atmosphere or random effects in an observing system. This component is also known as *stochastic* or *irreducible* uncertainty, since no deterministic model can anticipate the effect of random processes.

- *Epistemic* comes from the Greek word *epistēmē*, which means knowledge. *Epistemic uncertainty in mathematics* denotes all uncertainty that is not due to random processes, that is, all uncertainty from our *lack of knowledge* about the observed system. Since better knowledge—including improvements to the model and observing system—can reduce this uncertainty, it is also known as *reducible uncertainty*.

In contrast, ML approaches typically assume that 1) only the data are provided, with no information on the data-generation process; 2) the data cannot be changed. Based on this limitation, the ML literature draws the line between aleatory and epistemic uncertainty based on whether the uncertainty originates from the data, regardless of whether it is due to stochastic processes or lack of knowledge (Fig. 3). This difference between definitions can lead to great confusion. Henceforth, we use the terms ML-aleatory and ML-epistemic to refer to the ML version of those definitions.

To make matters worse, the ML literature sometimes uses alternate terms—*stochastic* or *irreducible uncertainty*—for the ML-aleatory definition, which is truly misleading. Hüllermeier and Waegeman (2021) nicely illustrate the consequences of using the term *irreducible* for ML-aleatory uncertainty: "This characterization, while evident at first sight, may appear somewhat blurry upon closer inspection. What does 'reducible' actually mean?" The impact of changing only the ML model

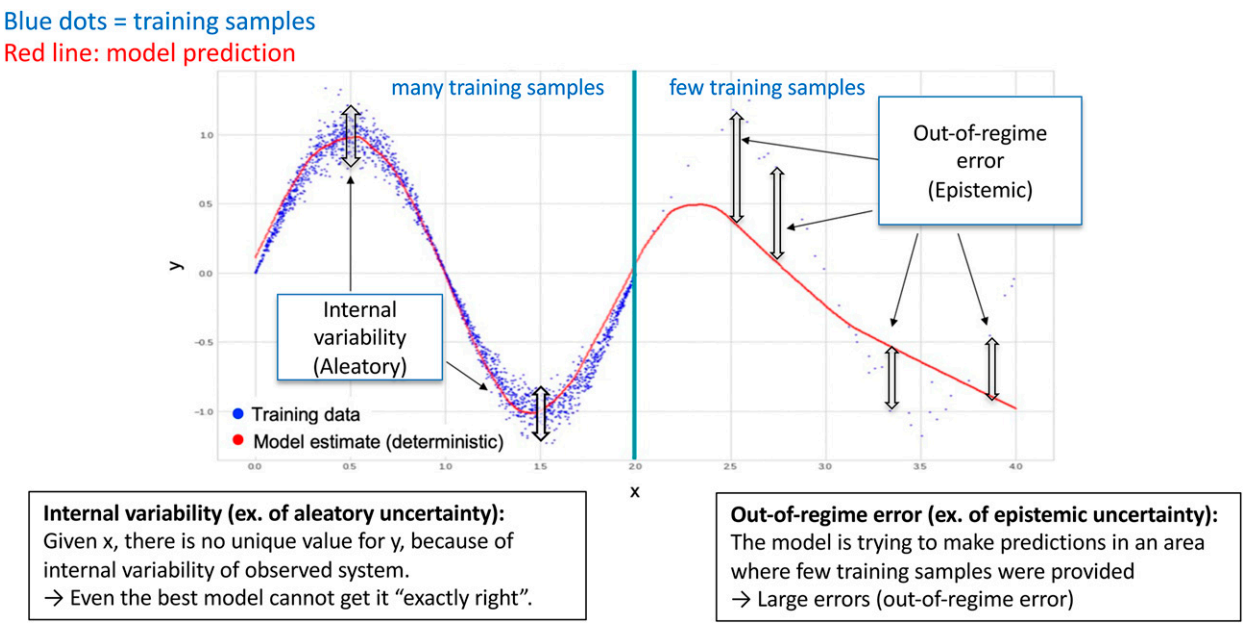


FIG. 2. Illustration of uncertainty due to internal variability vs out-of-regime error, based on synthetic data. The training dataset is shown as blue dots and has two issues: varying randomness in y (for all x) and varying sampling rate in x (for $x > 2$). Predictions from a deterministic NN are shown in red. Similar illustrations are often shown in ML to illustrate the concepts of aleatory vs epistemic uncertainty. Note that this is a highly idealized example, as it neglects to illustrate other types of aleatory and epistemic uncertainty. For this idealized example the definitions of aleatory and epistemic uncertainty from mathematics and ML align.

is obvious: a better ML model reduces epistemic uncertainty while leaving aleatory uncertainty unchanged, according to both the math and ML definitions. However, Hüllermeier and Waegeman (2021) point out that a change of the dataset can have less obvious consequences. Adding more predictors to the dataset[1]—without increasing sample size—is likely to decrease ML-aleatory uncertainty by increasing information about the system's state, but at the same time increasing ML-epistemic uncertainty, since the ML model might not be able to represent more complex relationships without additional data samples. In this scenario, the *supposedly irreducible* (ML-aleatory) uncertainty was *reduced*. How can this be? What appears to be a contradiction is really just a demonstration of why the terms *irreducible* and *stochastic* should be avoided as aliases for the ML-aleatory component. Finally, note that while the mathematical definition assigns any uncertainty *uniquely* to either the aleatory or epistemic type for a given system (at least in theory), the ML definition is context dependent. As demonstrated above, ML-aleatory uncertainty can become ML-epistemic, and vice versa, by a change of dataset.

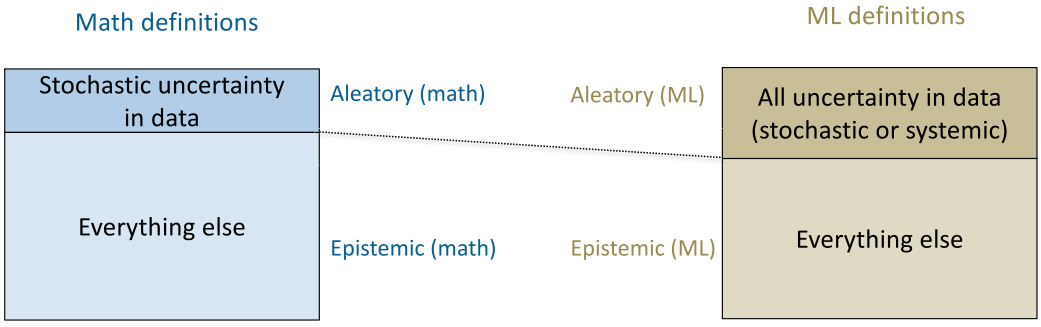
1) WHY DO WE CARE ABOUT DISTINGUISHING ML-ALEATORY VERSUS ML-EPISTEMIC UNCERTAINTY?

Knowing how much of the total uncertainty is ML-aleatory versus ML-epistemic can provide ML developers with valuable information on how to reduce uncertainty (Ortiz et al. 2022). If ML-aleatory uncertainty is high, we should focus on making the dataset more information-dense, for example, add predictors or improve data quality. If ML-aleatory uncertainty is low and ML-epistemic uncertainty is high, we should focus on improving the model. This can be done by changing the model structure or adding samples to the dataset; the latter reduces out-of-regime errors. Furthermore, understanding the difference between ML-aleatory and ML-epistemic uncertainty is crucial for understanding the limitations of UQ approaches, as many approaches cannot capture ML-epistemic uncertainty. As a general rule, non-Bayesian (maximum likelihood) approaches can capture only ML-aleatory uncertainty, while Bayesian approaches, such as MC dropout and BNNs, can capture both types (Dürr et al. 2020). Although Bayesian approaches *can* capture both types of uncertainty, it is unclear how well they do so in practice. This is a topic of active research—one more reason why it is crucial for environmental scientists to use/develop good practices for evaluating uncertainty estimates.

2) HOW CAN WE DISENTANGLE ML-ALEATORY AND ML-EPISTEMIC UNCERTAINTY?

Non-Bayesian methods can capture only ML-aleatory uncertainty, negating this issue. However, Bayesian methods can capture both ML-aleatory and ML-epistemic uncertainty. Ortiz et al. (2022) used a method to separate total-uncertainty estimates from a BNN into ML-aleatory and ML-epistemic components, demonstrating their method for precipitation-type retrieval from satellite data. We discuss Ortiz et al. (2022) further in section 3f.

[1] Assuming that the new predictors are not redundant with the preexisting predictors, i.e., that the new predictors contain new information.



alea: Latin word, referring to game of dice (random).
epistēmē: Greek word, meaning knowledge (model).

FIG. 3. The aleatory/epistemic divide differs among disciplines (Bevan 2022; Hüllermeier and Waegeman 2021). In the original math definition, the dividing line is whether the origin of uncertainty is stochastic. In the ML definition, the dividing line is whether the origin of uncertainty is in the given dataset. This is because in many ML applications the data are the only information available—i.e., we do not know about the data-generation process and have no ability to improve the dataset. However, environmental science contains many examples where neither of these conditions is true. The difference between the math and ML definitions can lead to great confusion.

c. *Representing and communicating uncertainty*

The estimated uncertainty in the y_{true} distribution can be represented in three different ways: the parameters of a canonical probability distribution, such as the normal distribution; nonparametric summary statistics (e.g., histogram bin frequencies or quantiles) of the y_{true} distribution; or an explicit ensemble, containing many samples from the y_{true} distribution. In the rest of this article, we refer to the estimated y_{true} distribution—regardless of which representation is used—as the y_{pred} distribution or *predicted distribution*.

It is possible to convert between different representations of the y_{pred} distribution. An ensemble can be converted to nonparametric summary statistics via simple equations, or to parameters of a canonical distribution via maximum-likelihood estimation. Parameters of a canonical distribution can be converted to an ensemble via sampling,[2] or to nonparametric summary statistics via closed-form equations.[3] It is more difficult to convert nonparametric summary statistics to other representations, and in this article we convert nonparametric summary statistics only to parameters of a canonical distribution, not to ensembles. Specifically, we convert quantiles to the mean and standard deviation of a normal distribution, using Eqs. (14) and (15) in Wan et al. (2014).

For visualization purposes, two common measures of uncertainty are the standard deviation and 95% confidence interval. For highly nonnormal distributions, showing additional measures—such as the skewness, a set of quantiles, or a subset of ensemble members—might also be useful. The best ways to communicate uncertainty with different users, such as forecasters and emergency managers, is currently a topic of active research (Rogers et al. 2023; Serr et al. 2023; Demuth et al. 2023).

[2] For the normal distribution, see numpy.random.normal in Python.
[3] For the normal distribution, scipy.stats.norm in Python.

3. Approaches for UQ in neural networks

We describe six popular UQ approaches: PDP, NPDP, EP, MM, MC dropout, and BNN. All six approaches can be used for both regression and classification. Although more UQ approaches exist, most environmental-science applications use some form or combination of the six listed. Applying these to real-world environmental-science applications, section 5 demonstrates PDP, EP, and MC dropout, and section 6 demonstrates NPDP and MC dropout.

The first three approaches are non-Bayesian and are summarized in Fig. 4. These approaches have been used extensively by the meteorology community, although mostly in a non-ML setting, to postprocess NWP ensembles. PDP assumes that the y_{true} distribution matches a canonical probability distribution $\mathscr{D}$ and estimates the parameters of $\mathscr{D}$; NPDP estimates summary statistics (e.g., the histogram or a set of quantiles) of the y_{true} distribution without assuming the form of the distribution; and EP generates many predictions to approximate the y_{true} distribution. Because these approaches are non-Bayesian and single model, they can capture only ML-aleatory, not ML-epistemic, uncertainty.

The multimodel approach (also non-Bayesian and summarized in Fig. 4) has been used to postprocess NWP ensembles but can be used to postprocess NN ensembles as well. If the individual NNs are trained with a deterministic loss function, the multimodel ensemble cannot capture ML-aleatory uncertainty. However, since the NNs are trained independently, the ensemble may capture ML-epistemic uncertainty.[4] The multimodel approach can be used in tandem

[4] For an out-of-regime data sample, because the NNs do not have enough training data to learn the true relationship, they are more sensitive to randomly initialized weights. Thus, for an out-of-regime sample, the NNs should produce very different predictions, leading to a larger spread for out-of-regime samples than for in-regime samples.

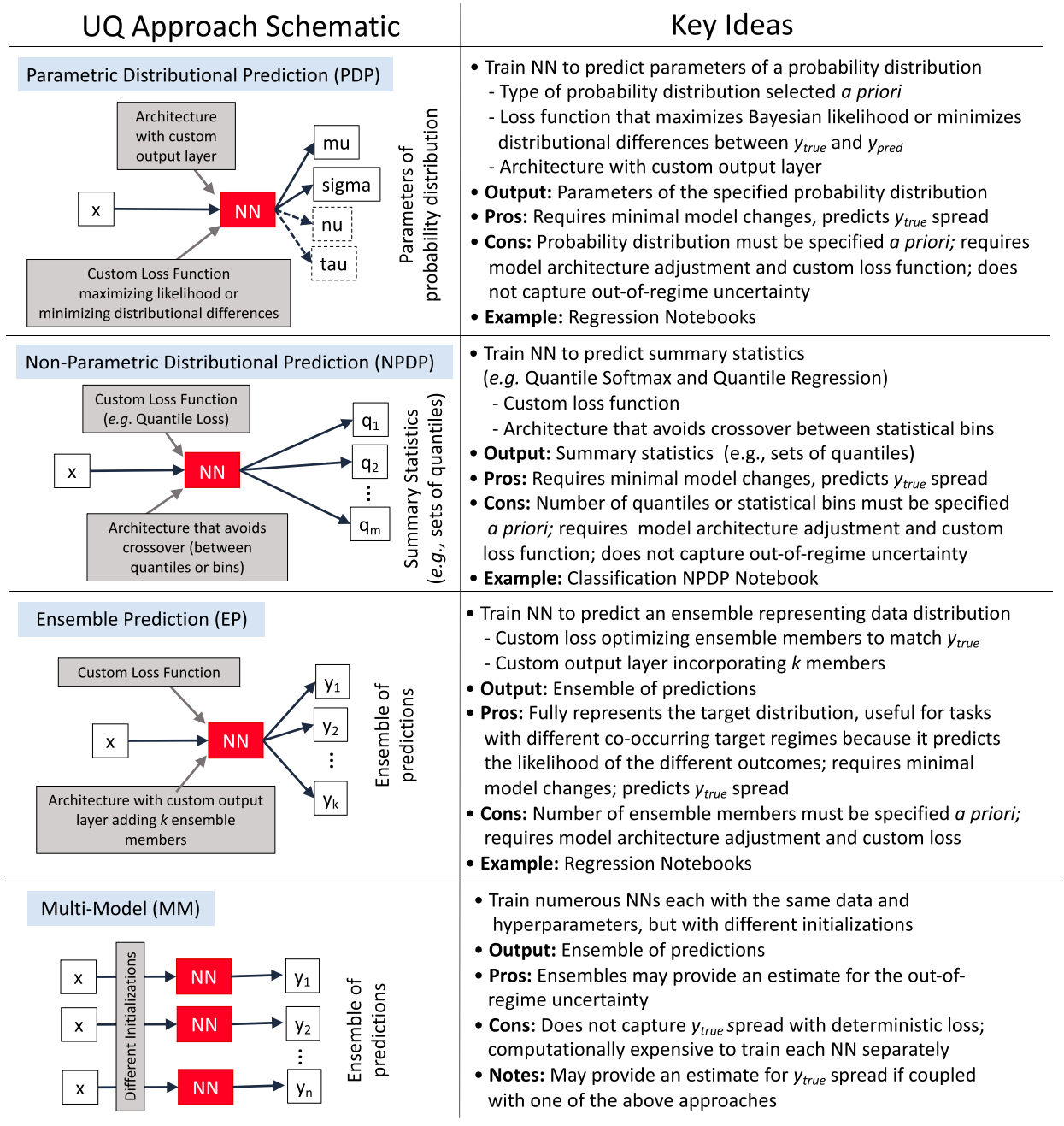


FIG. 4. Summary schematics and key ideas for non-Bayesian UQ approaches. These approaches have been used extensively by the meteorology community, although mostly in a non-ML setting, to postprocess NWP ensembles.

with any of the three UQ approaches, allowing both types of uncertainty to be captured.

The last two approaches are Bayesian techniques developed by the computer-science community specifically for NNs (Fig. 5). Both of these approaches—MC dropout and BNNs—use the NN itself to estimate ML-epistemic uncertainty, specifically by randomly selecting which model parameters (e.g., neuron weights) to use at inference time. When the NN is trained with a deterministic loss function, Bayesian approaches only capture ML-epistemic uncertainty; however, Bayesian approaches can easily be modified to incorporate any of the first three approaches, allowing them to capture both types of uncertainty.

a. PDP

PDP involves estimating the parameters of a canonical distribution, chosen a priori by the user. PDP can be used for both regression and classification, as long as the chosen

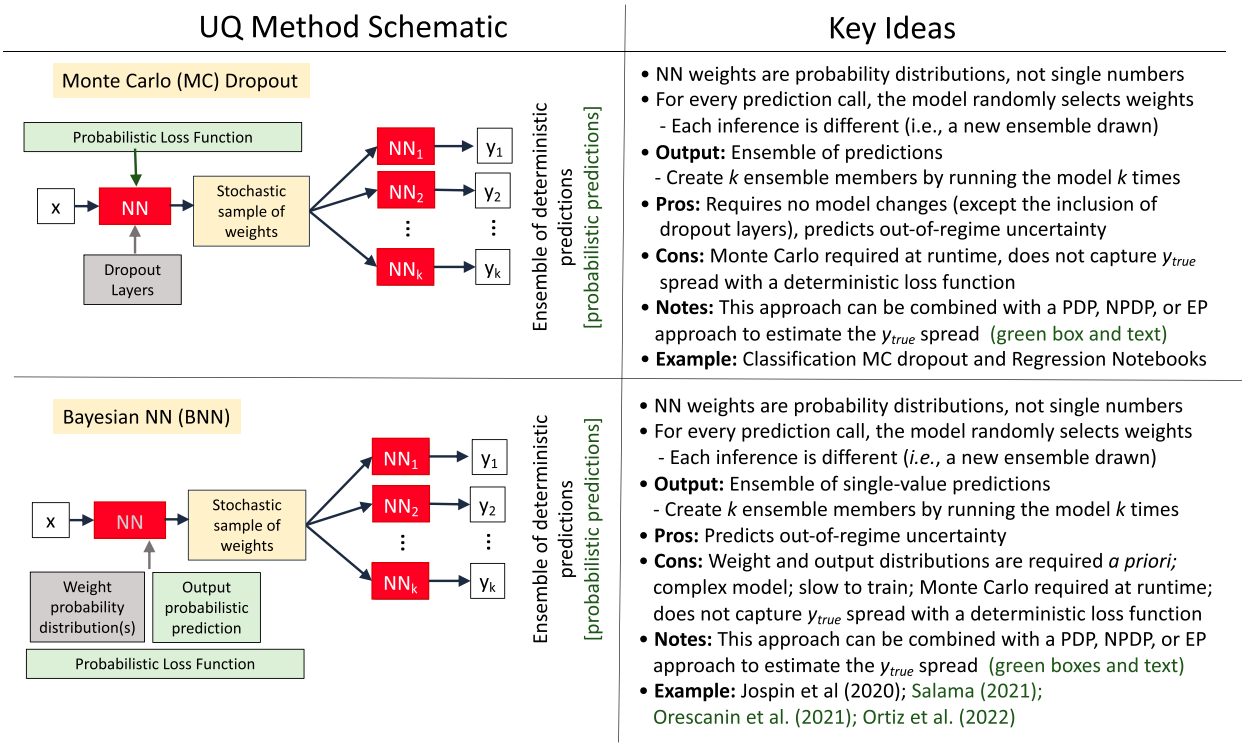


FIG. 5. Summary schematics and key ideas for Bayesian UQ approaches, developed by the computer-science community specifically for NNs.

distribution makes sense for the problem type.[5] A popular choice is the normal distribution, which has two parameters: the mean and standard deviation. In this case the NN estimates the mean and standard deviation for each data sample, with the standard deviation representing the NN's uncertainty.

There are two popular optimization methods for PDP. One is the maximum-likelihood approach, where parameters are chosen to maximize the probability of the observed data given the fitted distribution (Van Schaeybroeck and Vannitsem 2015; Bremnes 2020; Veldkamp et al. 2021; Schulz and Lerch 2022). As an example of this method, Barnes et al. (2021) used NNs with the sinh–arcsinh (SHASH) distribution for a problem with synthetic climate data. They adjusted the four distribution parameters—location, scale, skewness, and tail weight—to minimize the negative-log-likelihood loss function. Another method is to choose parameters that minimize distributional differences, most often using a closed form of the CRPS (see section 4e). Some examples are Van Schaeybroeck and Vannitsem (2015), Rasp and Lerch (2018), Baran and Baran (2021), Ghazvinian et al. (2021), Chapman et al. (2022), and Schulz and Lerch (2022). Studies comparing the two methods mostly favor the CRPS over the maximum-likelihood approach (Van Schaeybroeck and Vannitsem 2015; Table 5 of Veldkamp et al. 2021; Table 2 of Schulz and Lerch 2022). For Python code implementing PDP, see the regression notebooks.

[5] For example, the normal distribution should not be used for a classification problem, because it would allow class probabilities outside the range [0, 1].

Figure 6 shows sample results for PDP with NNs, one using the normal distribution and one using the SHASH distribution. The synthetic dataset contains one predictor x and one target y, with asymmetry and heteroskedasticity (Fig. 6a). Both NNs skillfully predict the mean, small uncertainty for x values with small spread, and large uncertainty for x values with large spread (Figs. 6b,c). For $x = 1$, where the marginal distribution (y given $x = 1$) is nearly normal, the two NNs have similar performance (Fig. 6d). For $x = 3$, where the marginal distribution is flatter than normal but still nearly symmetric, the two NNs again have similar performance (Fig. 6e). However, for $x = 7$, where the marginal distribution is highly skewed, the SHASH NN clearly outperforms the normal NN (Fig. 6f). This result highlights the advantage of a more flexible distribution. Figures 6b and 6c also highlight the advantage of a more flexible distribution around $x = 7$, where the SHASH NN captures the asymmetry and large spread better than the normal NN. We note that the purpose of this comparison is not to suggest that SHASH is the best distribution for all applications or even a particular application; rather, our purpose is to highlight the advantages of a more flexible distribution, *such as* SHASH, over the normal distribution.

b. *NPDP*

Two common NPDP methods are quantized softmax (QS) and quantile regression (QR). QS is used in several atmospheric-science applications (Wimmers et al. 2019; Scheuerer et al. 2020; Veldkamp et al. 2021) and involves turning a regression

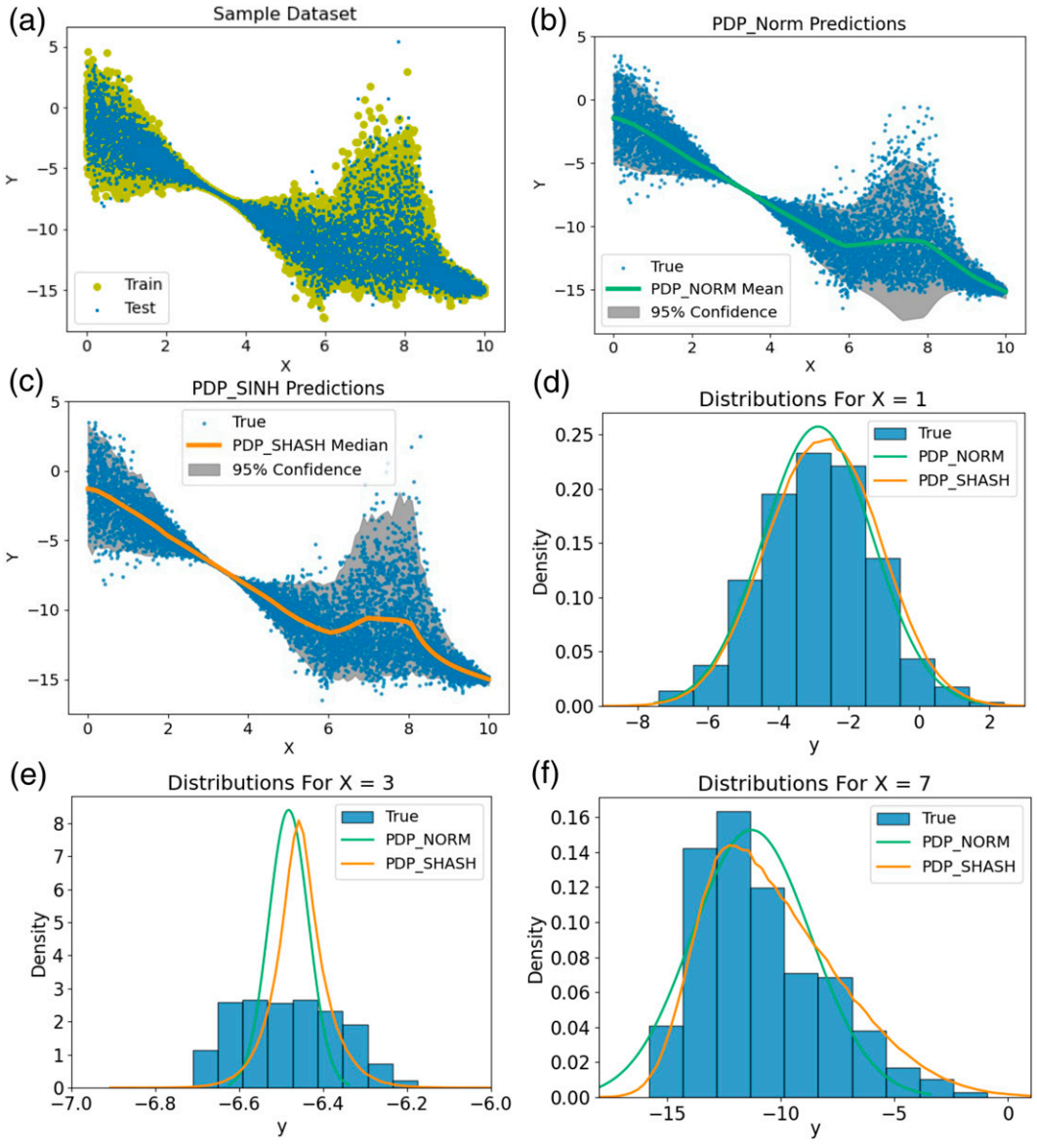


FIG. 6. Results for two NNs trained with different probability distributions. One NN uses a normal distribution, and one uses a SHASH distribution. (a) Scatterplot of the data. (b) Observations and predictions from normal NN. (c) Observations and predictions from SHASH NN. (d) Marginal distribution of observations and predictions from both NNs for $x = 1$. (e) As in (d), but for $x = 3$. (f) As in (e), but for $x = 7$.

problem into a classification problem via the following procedure:

1) Quantize the target variable y into K mutually exclusive and collectively exhaustive (MECE) bins. For example, if y is radar reflectivity (dBZ), the bins could be <0; [0, 1); [1, 2); …; [74, 75); and ≥ 75.
2) Each bin is considered one class, and the NN performs classification, using the softmax activation function. Softmax (section 6.2.2.3 of Goodfellow et al. 2016) ensures that the NN's K confidence scores all range from [0, 1] and sum to 1.0, allowing them to be interpreted as probabilities of the MECE classes.

QR involves directly predicting several quantiles of a probability distribution; it can be used for both regression and classification. QR has recently gained wide popularity for NNs (Bremnes 2020; Yu et al. 2020; Schulz and Lerch 2022). The "trick" is to train the NN with the quantile loss function:

$$\mathscr{L} = \begin{cases} q|y_{\text{true}} - y^q_{\text{pred}}|, & \text{if } y_{\text{true}} > y^q_{\text{pred}}; \\ (1-q)|y_{\text{true}} - y^q_{\text{pred}}|, & \text{if } y_{\text{true}} \leq y^q_{\text{pred}}. \end{cases} \quad (1)$$

The desired quantile level is $q \in [0, 1]$, and y^q_{pred} is the estimated value at quantile level q. Large values of q penalize underprediction ($y^q_{\text{pred}} < y_{\text{true}}$) more than overprediction ($y^q_{\text{pred}} > y_{\text{true}}$),

encouraging the model to output large y^{q}_{pred}. Conversely, small values of q encourage the model to output small y^{q}_{pred}.

To estimate multiple quantiles, a common approach is to train a separate NN for each quantile. However, because the different NNs are trained independently, this approach does not prevent quantile crossing, where y^{q}_{pred} decreases with q (e.g., the 25th percentile rainfall prediction is 30 mm, but the 75th percentile prediction is 20 mm). Two approaches have been developed to prevent quantile crossing with NNs. Bremnes (2020) averaged the predictions for each quantile level over 10 NNs, each trained with different random initializations and thus converging to different solutions. Without model averaging, quantile crossing occurred for 0.22% of cases; with model averaging, quantile crossing never occurred in their dataset. However, the method of Bremnes (2020) is not guaranteed to prevent crossing. Schulz and Lerch (2022) used an NN to predict Bernstein polynomials, which were transformed to quantiles in a postprocessing step. They noted that if the Bernstein coefficients do not cross—that is, are monotonically nondecreasing with q—then the y^{q}_{pred} also do not cross. Thus, Schulz and Lerch (2022) trained the NN to predict the *increment* between Bernstein coefficients for each pair of consecutive q values, using the softplus activation function to ensure that each increment is nonnegative. We note that quantile-regression forests (Meinshausen and Ridgeway 2006) prevent quantile crossing by default, but the current work focuses on NNs instead of random forests.

Our solution is similar to Schulz and Lerch (2022), except that we encode all logic directly in the NN architecture. Thus, our NN outputs are already quantile-based estimates, with no need for postprocessing. To satisfy the monotonicity constraint—namely, that $y^{q_i}_{\text{pred}} \geq y^{q_{i-1}}_{\text{pred}}$ for quantile levels $q_i > q_{i-1}$—we express $y^{q_i}_{\text{pred}}$ as the sum of $y^{q_{i-1}}_{\text{pred}}$ and a nonnegative term. Specifically, we implement the following equation:

$$y^{q_i}_{\text{pred}} = y^{q_{i-1}}_{\text{pred}} + \text{ReLU}(\Delta y^{q_i}_{\text{pred}}), \tag{2}$$

where ReLU is the rectified linear unit (Nair and Hinton 2010), defined as $\text{ReLU}(w) = \max(0, w)$.

To implement Eq. (2) inside the NN, we allow the NN to estimate $\Delta y^{q_i}_{\text{pred}}$ freely, use a ReLU layer to make this increment nonnegative, then use an Add layer to achieve the addition on the right-hand side. (For a schematic representation of this procedure, see Fig. 17b and the accompanying discussion in section 6a.) For Python code, see the NPDP notebook for classification.

c. Ensemble prediction

In this approach, a single NN is trained to produce an ensemble that captures the spread in y_{true}. Each output neuron corresponds to one ensemble member. To encourage appropriate spread in the ensemble, the NN must be trained with a custom loss function that minimizes distributional differences (i.e., differences between the y_{true} and y_{pred} distributions). A common loss function for this approach is the CRPS (section 4e). For Python code that implements EP with the CRPS loss, see the regression notebooks.

d. Multimodel approach

The multimodel approach involves training several NNs, each with the same data and structure but a different initialization—that is, set of random initial weights. Each NN converges to a different solution—that is, set of final trained weights—yielding a different prediction for the same data sample. The ensemble size is the number of NNs.

Although quite popular in environmental science (e.g., Doblas-Reyes et al. 2005; DelSole et al. 2013; Beck et al. 2016), the multimodel approach has two major disadvantages. First, because the NNs are trained independently and not optimized to produce good uncertainty estimates, they often perform poorly. Second, the multimodel approach is computationally expensive, because several NNs, instead of just one, must be trained and then loaded at inference time. We note that any other UQ method can be combined with the multimodel approach—for example, QR (Bremnes 2020) or PDP (Rasp and Lerch 2018)—often leading to better results.

e. MC dropout

Dropout regularization (or just "dropout") was invented to prevent overfitting in NNs (Hinton et al. 2012). During each forward pass through the NN, a random subset of neurons is dropped out, leaving the remaining neurons to represent useful features of the predictor data. Because the remaining neurons must still be able to represent predictor features adequately, dropout forces all neurons to learn more independently of each other, creating a pseudoensemble. In common practice, dropout is used only during training; at inference time all neurons are used, making the NN deterministic. However, dropout can also be used at inference time, making the NN stochastic and producing a predicted *distribution* by running the NN many times, which is called MC dropout (Gal and Ghahramani 2016).

The main advantage of MC dropout is ease of implementation. Keras has a predefined dropout layer,[6] and passing the argument training = true during model construction ensures that dropout will be used at inference time, as shown in the MC and regression notebooks. However, MC dropout has three major disadvantages. First, sampling (i.e., running the NN many times in inference mode) is computationally expensive and the correct hyperparameters (which dropout rate to use in each layer) are unclear. Second, although MC dropout captures out-of-regime uncertainty (a type of ML-epistemic uncertainty) well, it does not capture ML-aleatory uncertainty well (Bihlo 2021; Klotz et al. 2022; Garg et al. 2022). However, MC dropout can be combined with postprocessing methods to provide more holistic uncertainty estimates (Sato et al. 2021; Yagli et al. 2022). Third, MC dropout often performs poorly. This is because MC dropout performs UQ in a post hoc manner, using a regularization method designed to be turned off at inference time. Thus, one *hopes* that the model will produce good uncertainty estimates, without

[6] Other NN libraries, such as PyTorch, also have a Dropout layer. However, we have not experimented with dropout at inference time in these libraries.

directly optimizing it to do so. In fact, we are unaware of an atmospheric-science application where MC dropout performs better than another UQ method. However, due to its ease of implementation, MC dropout is still popular and can easily be used as a baseline to compare against other UQ methods.

MC dropout can be used simultaneously with any of the other UQ approaches, allowing the model to capture both ML-epistemic and ML-aleatory uncertainty. To achieve this, simply add dropout to the desired layers in the original NN, passing the argument training = true to ensure that dropout will be used at inference time.

f. BNN

BNNs are gaining popularity as a UQ approach, because they can capture both ML-aleatory and ML-epistemic uncertainty; however, they are conceptually and computationally complex. Although MC dropout is also a Bayesian method (Gal and Ghahramani 2016), BNNs are more flexible and may provide more robust uncertainty estimates (Salama 2021; Jospin et al. 2022). BNNs require both a functional model (a traditional NN) and a stochastic model; during training, they use Bayesian inference to sample from the stochastic model. A traditional NN learns one value for each model parameter, but a BNN learns a full *distribution* for each model parameter, determined by fitting a canonical distribution such as the normal. When trained with a deterministic loss function, BNNs capture only ML-epistemic uncertainty, like the MC-dropout approach used in this work. However, also like MC dropout, BNNs can be combined with PDP, NPDP, or EP to capture ML-aleatory uncertainty as well (see Fig. 5 bottom).

For a detailed tutorial on BNNs, see Jospin et al. (2022). In the atmospheric-science literature, Orescanin et al. (2021) and Ortiz et al. (2022) used BNNs to classify precipitation type from satellite data. Orescanin et al. (2021) covers a more basic implementation of BNNs; Ortiz et al. (2022) expands on the first work by combining the BNN with PDP, allowing it to capture ML-aleatory uncertainty, and introducing a method to separate the BNN's total-uncertainty estimate into ML-aleatory and ML-epistemic components. They make 25 predictions per data sample—each with a different set of NN parameters, sampled from the fitted distribution of weights for each neuron—yielding a 25-member ensemble. Ortiz et al. (2022) show that 1) BNNs can provide skillful mean predictions and uncertainty estimates; 2) decomposing the uncertainty estimates allows users to make informed decisions, not only on how to best use the model's predictions, but also on how to improve the whole ML pipeline, from data collection and data processing to model training and hyperparameter tuning.

Although BNNs can be powerful tools for UQ, their practical utility is still being investigated. BNNs have more model parameters than traditional NNs—K weights per neuron for a K-parameter canonical distribution, rather than one weight per neuron. Also, training a BNN involves Bayesian inference, which is computationally expensive and often fails due to memory limitations (Sato et al. 2021). Due to their complexity, we will not use BNNs in the case studies presented here, leaving this as future work and referring readers to the cited literature.

4. Methods for evaluating uncertainty estimates

This section discusses graphics and scoring metrics that are useful for evaluating uncertainty estimates. The graphics allow for deeper insight than scoring metrics (single-number summaries), while the scores allow for easy comparison of different methods/models. One of the evaluation graphics presented—the attributes diagram—pertains to the central (mean) prediction, while the others pertain to uncertainty estimates. We include the attributes diagram because, even if uncertainty is the main focus, evaluating the main predictions is important as well. If the mean predictions are poor, then 1) the uncertainty estimates are likely to be poor as well; 2) even if the uncertainty estimates are skillful, they are of little use. All methods discussed in this section are summarized in Table 1.

All evaluation methods require summary statistics from the y_{pred} distribution: the mean, standard deviation, cumulative distribution function (CDF) at one y_{true} value, or probability distribution function (PDF) at one y_{true} value. As discussed in section 2c, the mean and standard deviation can easily be obtained from any y_{pred} distribution, regardless of how it is represented. Meanwhile, the CDF can easily be obtained from parameters of a canonical probability distribution[7] or from an ensemble.[8] The CDF can also be obtained from nonparametric summary statistics like quantiles, as the quantile function is the inverse CDF. Last, the PDF can be easily obtained from the CDF via differentiation.

We demonstrate all evaluation methods on a regression task with a synthetic dataset (Fig. 6a), using four UQ methods that cover three UQ approaches (PDP, EP, and MC dropout). For PDP, we use two different canonical distributions, normal and SHASH; for EP, we use the CRPS loss function (see section 4e for details); and for MC dropout, we use the MSE loss function. Since we use a deterministic loss function, we do not expect MC dropout to capture ML-aleatory uncertainty. We use the same synthetic dataset to illustrate all evaluation tools except the CRPS score (Fig. 12), where we use a dataset with a bimodal distribution to illustrate the advantages of the EP-CRPS method. All eight metrics for all four UQ methods are summarized in Table 2, and these results can be reproduced in the regression notebook for model comparison.

a. The attributes diagram

1) FOR CLASSIFICATION TASKS

The attributes diagram (Fig. 7a) is a reliability curve with additional elements. The reliability curve plots model-predicted

[7] For the normal distribution, see scipy.stats.norm.cdf in Python.

[8] See statsmodels.distributions.empirical_distribution.ECDF in Python.

TABLE 1. Evaluation methods for the mean prediction (attributes diagram) and uncertainty estimates (all others). A check mark under "Class?" indicates whether the evaluation method can be used for classification models, and "Reg?" indicates whether it can be used for regression models. Evaluation graphics are in bold; evaluation scores are in italics. All methods are demonstrated in the regression notebooks; if demonstrated in additional notebooks, these are included in square brackets.

Method	Class?	Reg?	What it tells us
Attributes diagram	✓	✓	Class: conditional bias (i.e., as a function of predicted event probability), BSS Reg: conditional bias, MSESS Ideal plot follows the 1-to-1 line (no conditional bias).
BSS	✓		Brier score improvement over climatology. BSS $>$ 0 means improvement; BSS = 1 means perfect model.
MSESS		✓	See above, replacing "Brier score" with "MSE."
Spread–skill plot	✓	✓	RMSE of mean prediction as a function of y_{pred} spread (i.e., standard deviation). Ideal plot follows the 1-to-1 line (spread = RMSE). [classification notebooks]
SSRAT	✓	✓	Spread/RMSE averaged over dataset. Ideal value = 1.
SSREL	✓	✓	Weighted distance between spread–skill plot and 1-to-1 line. Ideal value = 0.
Discard test	✓	✓	Model error vs discard fraction. Error should decrease whenever discard fraction is increased. [classification notebooks]
MF	✓	✓	*How often* error decreases when discard fraction is increased. Ideal value = 1.
DI	✓	✓	*How much* error decreases on average when discard fraction is increased. Higher values are better.
PIT histogram		✓	Distribution of PIT values. Ideal histogram is flat, indicating a uniform distribution.
PITD		✓	PIT calibration-deviation score. Mean difference between actual bin frequency and expected bin frequency for uniform histogram. Ideal value = 0.
CRPS	✓	✓	Mathematically: area between predicted and observed CDFs. Conceptually: how well y_{pred} distribution captures y_{true} spread. Ideal value = 0.
IGN	✓	✓	How much y_{pred} distribution is concentrated in the correct areas. Rewards narrow y_{pred} distribution containing the observation. Ideal value = 0.

event probability[9] on the x axis versus conditional observed event frequency on the y axis.[10] Each point corresponds to one bin of event probabilities. For example, suppose that the event is tornado occurrence; the reliability curve uses 10 probability bins, equally spaced from 0.0 to 1.0; and one point on the curve is (0.15, 0.4). Hence, in cases where the model predicts a tornado probability between 10% and 20%, a tornado actually occurs 40% of the time. The reliability curve is used to identify conditional model bias, that is, bias as a function of the event probability. Labels in Fig. 7a show how to identify conditional model bias from the reliability curve—that is, for which event probabilities the model has positive bias (overconfident), negative bias (underconfident), or zero bias.

The full attributes diagram is a reliability curve with four additional elements: the 1-to-1 line, no-resolution line, climatology line, and positive-skill area. The last is a polygon defining where the Brier skill score (BSS) is positive. The BSS is defined as $(\text{BS}_{\text{climo}} - \text{BS})/\text{BS}_{\text{climo}}$, where BS and BS_{climo} are the Brier scores of the model of interest and the climatological model. The BSS ranges from $(-\infty, 1]$, and values > 0 signal an improvement over climatology. For more details on the attributes diagram, see Hsu and Murphy (1986).

2) FOR REGRESSION TASKS

Although the attributes diagram is typically used for classification, it can also be adapted for regression (Fig. 7b). Letting the target variable be z, the x axis is the predicted z value and the y axis is the conditional mean observed z value, both real

TABLE 2. Evaluation scores for various UQ methods on the synthetic dataset. For each score, the best value is highlighted in bold.

Score	PDP_NORM	PDP_SHASH	EP_CRPS	MC_DROPS
MSESS	0.847	0.844	**0.849**	0.843
SSRAT	0.769	0.759	**0.789**	0.363
SSREL	**0.079**	0.110	0.114	0.993
MF	**1.000**	**1.000**	**1.000**	0.300
DI	0.185	**0.187**	0.176	0.019
PITD	0.013	**0.006**	0.008	0.064
CRPS	0.748	0.737	**0.717**	0.892
IGN	**1.678**	1.728	1.775	4.886

[9] When ML models are used for classification, by default they produce confidence scores ranging from [0, 1], which are not true probabilities. However, in this paper we adopt common parlance and refer to confidence scores as probabilities.

[10] The reliability curve uses only the central prediction—i.e., the x coordinate is the mean of the predicted distribution, not a measure of uncertainty. However, UQ studies commonly employ the reliability curve to evaluate the central prediction (e.g., Delle Monache et al. 2013; Jospin et al. 2022; Chapman et al. 2022), so we include it in this paper.

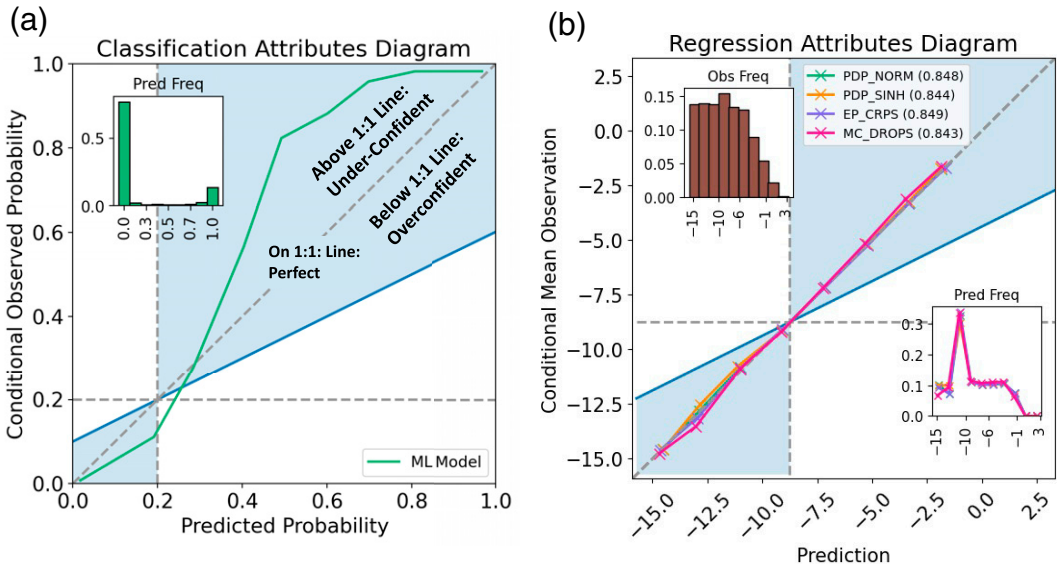


FIG. 7. (a) Attributes diagram for classification, where the relative frequency of occurrence of the event is 0.2. The diagonal, horizontal, and vertical gray dashed lines are the 1-to-1, no-resolution, and climatology lines, respectively; the blue-shaded polygon is the positive-skill area; and the green line is the NN's reliability curve. Points below, above, and on the 1-to-1 line correspond to probabilities where the NN is overconfident, underconfident, and perfectly calibrated, respectively. The inset histogram shows the full distribution of $\overline{y_{\text{pred}}}$, the NN's mean predicted probability. (b) Attributes diagram for regression, where the true mean of the data is −8.76. Reliability curves are shown for four NNs; the MSESS for each NN is shown in the legend. The inset histogram at the top left shows the full distribution of y_{true}, and the inset histogram at the bottom right shows the full distribution of $\overline{y_{\text{pred}}}$; for a perfect NN the two histograms would match. All other elements of (b) are as in (a).

numbers that in general can range from $(-\infty, +\infty)$. The positive-skill area shows where the MSESS, defined analogously to the BSS, is positive. Otherwise, the two flavors of attributes diagram can be interpreted the same way.

b. *The spread–skill plot*

The spread–skill plot (Delle Monache et al. 2013) is analogous to the reliability curve but evaluates uncertainty estimates rather than mean predictions. Conceptually, the spread–skill plot answers the question: "For a given predicted model spread, what is the actual model error?" The spread–skill plot shows the predicted model spread (x axis) versus the actual model error (y axis), as in Fig. 8.

Specifically, the x axis is the mean standard deviation ($\overline{\text{SD}}$) of the model's predicted distribution, while the y axis is the root-mean-squared error (RMSE) of the model's mean prediction. Each point corresponds to one bin of spread values. The two quantities are defined as follows for the kth bin:

$$\begin{cases} \text{RMSE}_k = \left[\dfrac{1}{N_k}\displaystyle\sum_{i=1}^{N_k}(y_i^{\text{true}} - \overline{y_i^{\text{pred}}})^2\right]^{1/2}; \\ \overline{\text{SD}_k} = \dfrac{1}{N_k}\displaystyle\sum_{i=1}^{N_k}\left[\dfrac{1}{M-1}\sum_{j=1}^{M}(\overline{y_i^{\text{pred}}} - y_{ij}^{\text{pred}})^2\right]^{1/2}; \\ \overline{y_i^{\text{pred}}} = \dfrac{1}{M}\displaystyle\sum_{j=1}^{M} y_{ij}^{\text{pred}}. \end{cases} \tag{3}$$

The term y_i^{true} is the observed value for the ith example; $\overline{y_i^{\text{pred}}}$ is the mean prediction for the ith example; y_{ij}^{pred} is the jth prediction for the ith example; N_k is the total number of examples in the kth bin; and M is the ensemble size.[11]

Labels in Fig. 8 show how to identify conditional bias in the model's uncertainty estimates—that is, for which spread values the model is underdispersive (overconfident), overdispersive (underconfident), or perfectly calibrated (spread–skill ratio = 1). Additionally, if the spread frequency is most-populated where the points on the spread–skill diagram lie on the 1:1 line, then the uncertainty estimates are useful—for example, the samples with the lowest (highest) spread have corresponding low (high) error.

The quality of the spread–skill plot can be summarized by two measures: SSREL and overall SSRAT. SSREL is the weighted mean distance from the 1-to-1 line:

$$\text{SSREL} = \sum_{k=1}^{K}\frac{N_k}{N}\left|\text{RMSE}_k - \overline{\text{SD}_k}\right|, \tag{4}$$

where N is the total number of examples; K is the total number of bins; and other variables are as in Eq. (3). SSREL varies from [0, ∞), and the ideal value is 0. Meanwhile, SSRAT is the spread–skill ratio averaged over the whole dataset:

[11] As mentioned in section 2c, for UQ approaches that do not produce an ensemble, there is still a way to compute the standard deviation of the predicted distribution.

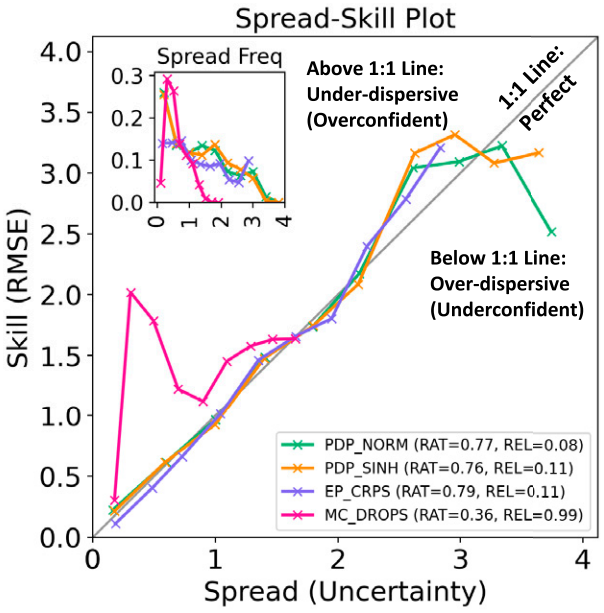


FIG. 8. Spread–skill plot. The inset histogram shows how often each spread value occurs. Points below, above, and on the 1-to-1 line correspond to spread values where the model is underconfident, overconfident, and perfectly calibrated, respectively.

$$\text{SSRAT} = \frac{\overline{\text{SD}}}{\text{RMSE}}, \tag{5}$$

where RMSE and $\overline{\text{SD}}$ are analogous to RMSE_k and $\overline{\text{SD}}_k$, respectively, but averaged over the whole dataset instead of one bin. SSRAT varies from [0, ∞), and the ideal value is 1. SSRAT > 1 indicates that the model is underconfident on average; SSRAT < 1 indicates that the model is overconfident on average.

The spread–skill plot and both summary measures can be used for both regression and classification. In the regression case, spread, skill, and SSREL are all in physical units. In the classification case, spread, skill, and SSREL are in units of class probability, ranging from [0, 1]. SSRAT is always unitless.

c. *The discard test*

The discard test, inspired by Barnes and Barnes (2021) and similar to the filter experiment in Fig. 8.18 of Dürr et al. (2020), compares model error versus the fraction of highest-uncertainty cases discarded. Sample results are shown in Fig. 9.

For a model with useful uncertainty estimates, the error should decrease whenever discard fraction is increased. The quality of the discard test can be summarized by two measures: MF and DI. The MF quantifies how often model error decreases—regardless of *how much* it decreases—when the discard fraction is increased:

$$\text{MF} = \frac{1}{N_f - 1} \sum_{i=1}^{N_f - 1} \mathscr{I}(\varepsilon_i \geq \varepsilon_{i+1}). \tag{6}$$

The term N_f is the number of discard fractions used; ε_i is the model error with the ith discard fraction, which is greater than

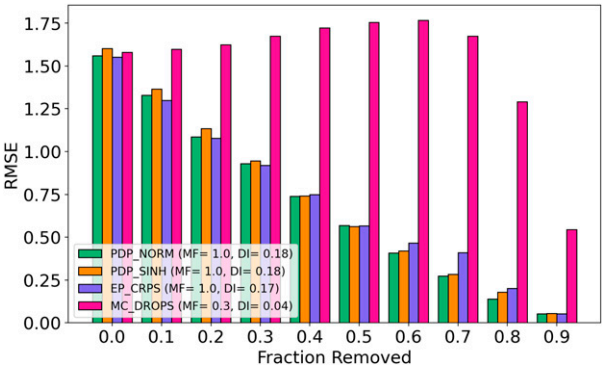


FIG. 9. Sample results for discard test. In this case, the error metric used is the RMSE of the NN's mean prediction. The legend shows the MF and DI for each NN.

the $(i - 1)$th discard fraction; and $\mathscr{I}()$ is the indicator function, evaluating to 1 if the condition is true and 0 otherwise. MF varies from [0, 1], and the ideal value is 1. Meanwhile, DI quantifies the average decrease in model error when discard fraction is increased:

$$\text{DI} = \frac{1}{N_f - 1} \sum_{i=1}^{N_f - 1} (\varepsilon_i - \varepsilon_{i+1}). \tag{7}$$

DI varies in general from (−∞, ∞), and higher values are better.

Neither summary measure tells the whole story. The MF is 1.0 (the ideal value) as long as error always decreases when the discard fraction is increased, even if the error decreases by a very small (insignificant) amount. Also, the DI involves the mean, which is strongly influenced by outliers. For example, in Fig. 9 the DI for MC dropout is strongly influenced by the large drop in RMSE as discard fraction is increased from 0.8 to 0.9. This large drop leads to a positive DI for MC dropout, even though model error usually *increases* when discard fraction is increased, the opposite of the desired effect. Thus, the MF and DI should always be used in tandem.

The discard test and both summary measures can be used for both regression and classification. In the regression case, ε_i is a regression-based error metric like RMSE; in the classification case, ε_i is a classification-based metric like cross entropy.

The discard test could be very useful in an operational environment. For example, consider results in Fig. 9 and suppose that there is an alternate prediction method to the NNs, which is true for most NN applications to environmental science. Also, suppose that the maximum acceptable error for this application is 1.0. Figure 9 shows that for three of the four NNs—all except the one using MC dropout—error < 1.0 for all discard fractions ≥ 30%. Thus, for each of these three NNs, an operational forecaster could find the spread value matching a discard fraction of 30%—let this be s^*—and use the NN predictions only when spread < s^*.

We note a crucial difference between evaluation methods. The discard test is concerned only with *ranking quality*. If the model correctly ranks uncertainty among all data samples,

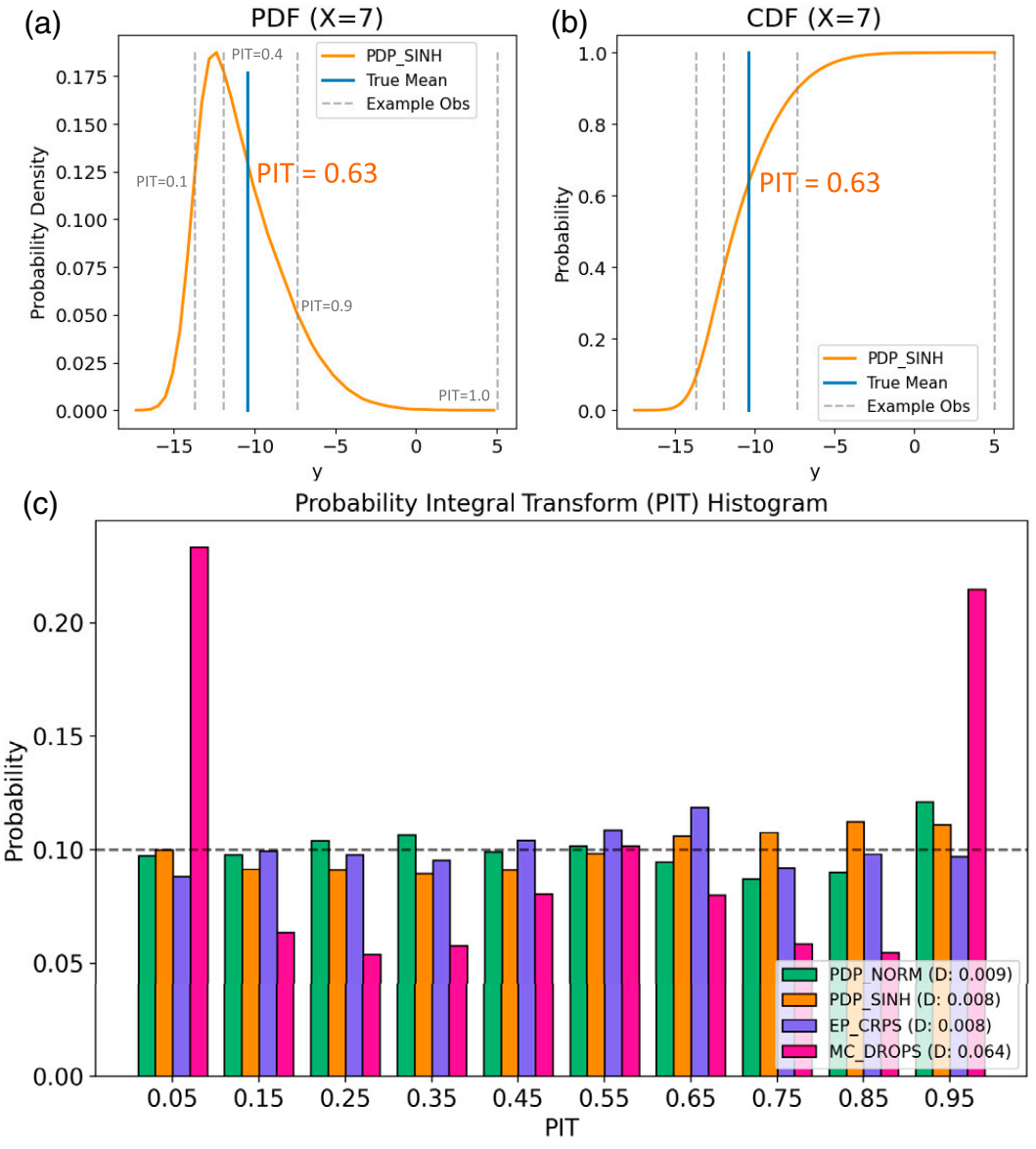


FIG. 10. (a),(b) Schematics explaining the PIT. These use the PDP_SINH model from Fig. 6, evaluated at $x = 7$. (a) PDF of predicted distribution, along with PIT values corresponding to a few possible y_{true} values. The actual y_{true} value has a PIT of 0.63. (b) As in (a), but showing CDF instead of PDF. (c) Example of PIT histogram. The dashed line represents a perfect PIT histogram.

MF = 1.0 (the ideal value), even if the model has a persistent bias, that is, it always underestimates or overestimates uncertainty. Meanwhile, the spread–skill plot is concerned only with *calibration quality* or bias. If the spread and skill are equal for all spread values, the plot follows the 1-to-1 line (ideal), even if the model cannot accurately rank the uncertainty of its own predictions.

d. The PIT histogram

The PIT is $F(y_{\text{true}})$, where F is the CDF of the predicted distribution. In other words, the PIT is the quantile of the predicted distribution at which the observed value occurs. A few examples are shown in Fig. 10b. Note that the PIT is meaningful only for regression problems, not for classification. For classification the only possible observations are 0 and 1, while predictions (event probabilities) must range from [0, 1]. Thus, y_{true} always occurs at one extreme of the predicted distribution, so the only possible PIT values are 0 and 1. Intermediate PIT values do not occur, which makes for a trivial PIT histogram.

The PIT *histogram* plots the distribution of PIT over many data samples, with one PIT value per sample (Fig. 10c). For a perfectly calibrated model, all PIT values occur equally often, so the histogram is uniform. If the histogram has a hump in

the middle, there are too many examples with intermediate PIT values (where y_{true} occurs near the middle of the predicted distribution) and too few examples with extreme PIT values (where y_{true} occurs near the end of the predicted distribution), so the extremes of the predicted distribution are on average too extreme. In other words, the predicted distribution is on average too wide, so the model is overspread or "underconfident." If the histogram has humps at the ends (as for MC dropout in Fig. 10c), the predicted distribution is on average too narrow, so the model is underspread or "overconfident." The PIT histogram is a generalization of the rank histogram (or "Talagrand diagram"), which is more familiar to atmospheric scientists (Hamill 2001) and can be interpreted the same way.

We note that a uniform PIT histogram is a *necessary but not sufficient* condition for calibrated uncertainty. For example, see Fig. 2 of Hamill (2001), where a nearly uniform rank histogram is produced by a combination of three miscalibrated predicted distributions: one with a positively biased mean prediction, one with a negatively biased mean prediction, and one with zero bias in the mean prediction but excessive spread. This example illustrates why using multiple evaluation tools is important. Biases in the mean prediction would average to zero and therefore not be captured by the attributes diagram, but all three miscalibrated distributions would generate excessive spread,[12] which would be captured by the spread–skill plot.

The quality of the PIT histogram can be summarized by PITD (Nipen and Stull 2011):

$$\text{PITD} = \left[\frac{1}{K}\sum_{k=1}^{N_k}\left(\frac{N_k}{N} - \frac{1}{K}\right)^2\right]^{1/2}, \tag{8}$$

where K is the number of bins; N is the total number of data samples; and N_k is the number of data samples in the kth bin. PITD varies from [0, 1], and the ideal value is 0.

e. The CRPS

The CRPS is commonly used in atmospheric science to evaluate probabilistic forecasts, that is, to compare a predicted distribution to an observation (Matheson and Winkler 1976; Hersbach 2000; Gneiting et al. 2005). The CRPS is a generalization of the mean absolute error (MAE) for probabilistic forecasts:

$$\text{CRPS}(F, y_{\text{true}}) = \int_{-\infty}^{\infty}[F(y_{\text{pred}}) - \mathscr{H}(y_{\text{pred}} - y_{\text{true}})]^2 dy_{\text{pred}}, \tag{9}$$

where y_{true} is the single observed value; F is the CDF of the predicted distribution; y_{pred}, the variable of integration, is one value in the predicted distribution; and $\mathscr{H}$ is the Heaviside step function, evaluating to 1 if $y_{\text{pred}} \geq y_{\text{true}}$ and 0 otherwise. Thus, Eq. (9) is the error between the predicted and observed CDF, the second of which is a step function (see Fig. 11).

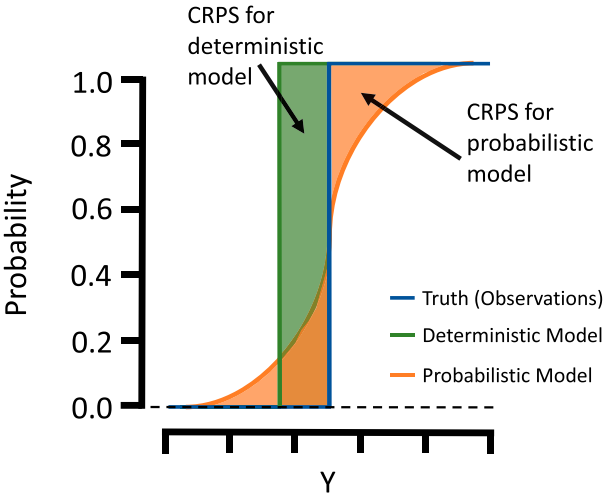


FIG. 11. Schematic representation of the CRPS. The axes show the CDF for y_{true}, $y_{\text{pred}}^{\text{determ}}$ (the single prediction from a deterministic model), and $\mathbf{y}_{\text{pred}}^{\text{prob}}$ (predictions from a probabilistic model). The CRPS for a given model is the area in the CDF plot between the y_{true} curve and the given model's y_{pred} curve. Adapted from Brey (2021).

Because the CRPS measures differences between the CDFs of y_{true} and y_{pred}, it can serve as a loss function for UQ in NNs. For datasets with uncertainty due to any of the sources discussed in section 2a, the conditional distribution $y_{\text{true}}|\mathbf{x}$ is no longer a point mass, so the corresponding CDF is no longer a step function. In this case the NN can be optimized to capture the spread in y_{true}, that is, to estimate the ML-aleatory uncertainty. To use the CRPS as a loss function, Gneiting and Raftery (2007) made the necessary modifications to Eq. (9), using distribution theory and identities from Székely and Rizzo (2005):

$$\text{CRPS} = \frac{1}{N}\sum_{i=1}^{N}|y_{\text{true}} - y_{\text{pred}}^{i}| - \frac{1}{2}\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}|y_{\text{pred}}^{i} - y_{\text{pred}}^{j}|, \tag{10}$$

where N is the ensemble size; y_{true} is the single observed value; and y_{pred}^{k} is the prediction for the kth ensemble member. The first term is the MAE of the mean prediction, and the second term is half the model spread, with "spread" defined as the mean absolute pairwise difference between ensemble members. This form of the CRPS can be used as a loss function for EP, and there are many examples in the atmospheric-science literature (Van Schaeybroeck and Vannitsem 2015; Scheuerer et al. 2020; Baran and Baran 2021; Dai and Hemri 2021; Ghazvinian et al. 2021; Veldkamp et al. 2021; Chapman et al. 2022; Schulz and Lerch 2022).

Figure 12 demonstrates the utility of the EP-CRPS approach. We use synthetic data with one predictor x and one target y, with a combined exponential and point-mass distribution. In other words, for some samples y increases

[12] Drawing y_{pred} values from the positively and negatively biased distributions would generate excessive spread, because their biases would ensure that most y_{pred} values are far above and below y_{true}, respectively.

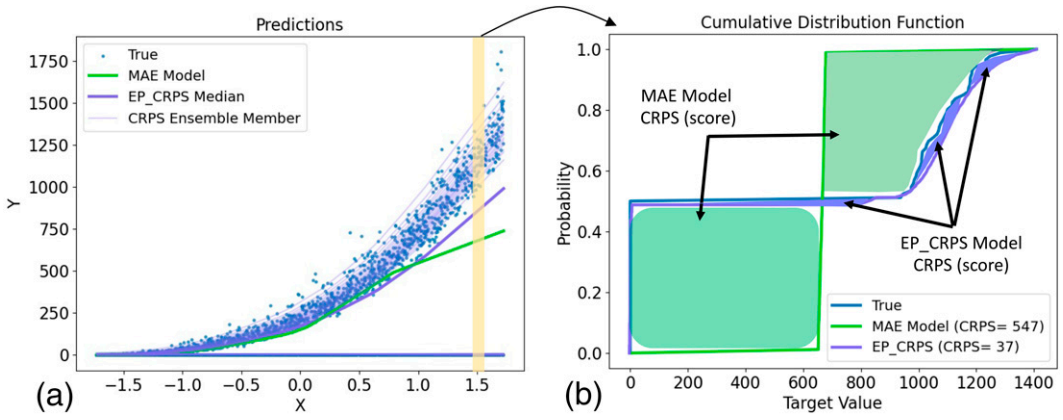


FIG. 12. Using two NNs to solve a simple problem (the relationship between target y and predictor x is described in the main text). One NN is deterministic and trained with the MAE loss; the other is probabilistic (EP) and trained with the CRPS loss. (a) Scatterplot of predictions and observations. For the CRPS-trained model, both individual ensemble members and the mean are shown. Note that many ensemble members are hidden below the observations at $y = 0$. (b)] Marginal CDFs (y given $x = 1.5$) for y_{true}, $y_{\text{pred}}^{\text{MAE}}$ (the single prediction of the MAE-trained model), and $\mathbf{y}_{\text{pred}}^{\text{CRPS}}$ (the predictions of the CRPS-trained model). As explained in the caption of Fig. 11, the CRPS for each model equals an area between two curves in this plot.

exponentially with x, and for some samples $y = 0$ regardless. Thus, for all x values except the smallest (~ -1.8), the marginal distribution (y given x) is bimodal, as shown in Fig. 12a. A model trained with the MAE loss function "splits the difference" between the two modes, always predicting poorly at higher x. Conversely, a model trained with the CRPS loss function produces ensemble members that capture both modes. Thus, the CRPS-trained model achieves a much better CRPS (37) than the MAE-trained model (547). However, the central prediction of the CRPS-trained model (heavy purple line in Fig. 12a) performs poorly, like the MAE-trained model. This serves as a reminder that the value of a probabilistic model is not only in the central prediction—one should consider the full predicted distribution. In addition to using the EP-CRPS approach to demonstrate the evaluation methods, we use it to solve a real-world regression problem (section 5).

The CRPS can also be modified to serve as a loss function for the PDP approach to UQ. Analytical forms of the CRPS have been derived for many canonical distributions. These include the normal distribution (Van Schaeybroeck and Vannitsem 2015; Rasp and Lerch 2018); lognormal, truncated lognormal, and truncated generalized extreme value (TGEV) distributions (Baran and Baran 2021); zero-truncated normal distribution (Chapman et al. 2022); censored, shifted gamma distribution (CSGD; Ghazvinian et al. 2021); and truncated logistic and piecewise uniform distributions (Schulz and Lerch 2022).

f. IGN

IGN measures how much a probabilistic forecast is concentrated in the correct areas (Good 1952; Roulston and Smith 2002; Nipen and Stull 2011). It is defined as

$$\text{IGN} = -\frac{1}{N}\sum_{i=1}^{N}\log_2[f(y_i^{\text{true}})], \tag{11}$$

where f is the PDF of the predicted distribution and y_i^{true} is the observed value for the ith data sample. Thus, $f(y_i^{\text{true}})$ is the predicted PDF evaluated at y_i^{true}. IGN varies from $[0, \infty)$, and the ideal value is 0. IGN rewards correct high-confidence predictions, that is, narrow predicted distributions that contain the observed value.

g. Score comparisons

Table 2 shows all eight scores discussed in section 4 for the four NNs—each with a different UQ method—applied to the synthetic dataset. All four NNs produce skillful mean predictions, indicated by an MSESS $\gg 0$. Also, mean predictions from the four NNs have nearly equal quality, indicated by the narrow range for MSESS (from 0.843 to 0.849). However, in terms of uncertainty, MC_DROPS clearly performs worse than the other UQ methods, achieving the worst value for all scores other than MSESS. As expected, because this model was trained with a deterministic loss function, it does not capture ML-aleatory uncertainty. The other three UQ methods—PDP with the normal distribution (PDP_NORM), PDP with the SHASH distribution (PDP_SHASH), and EP with the CRPS loss function (EP_CRPS)—achieve similar performance. PDP_NORM is the best-ranking method on two scores; PDP_SHASH ranks best on two scores; EP_CRPS ranks best on three scores; and the three methods share the best ranking on one score (MF).

5. Demonstration of UQ and evaluation methods for a regression task

a. Predicting dewpoint profiles for severe weather nowcasting

Vertical profiles of dewpoint are useful in predicting deep convection (i.e., thunderstorms), which can produce severe weather. Thunderstorms pose a lightning threat and may

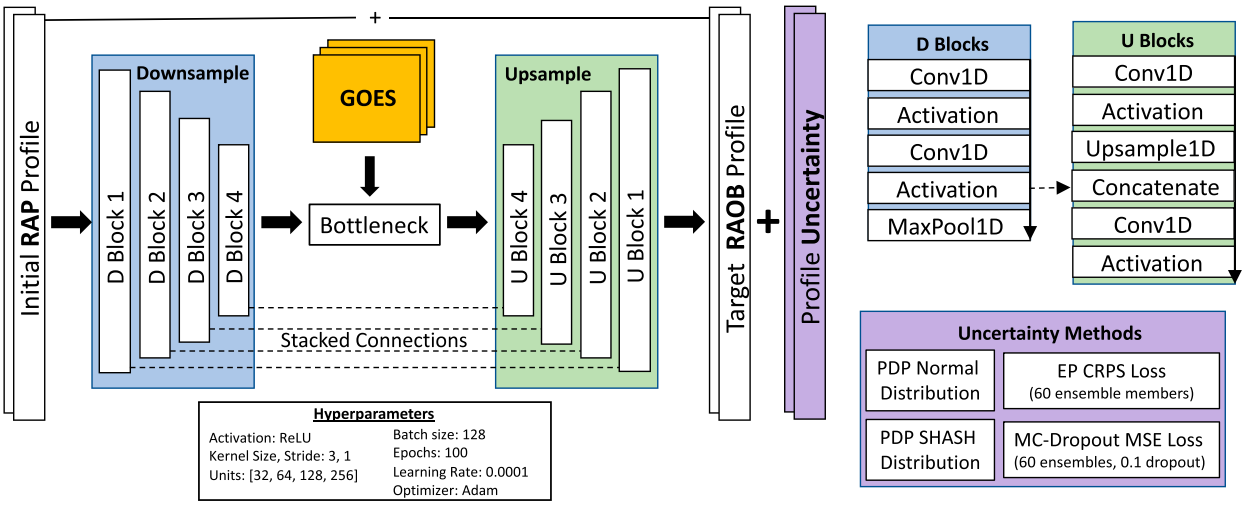


FIG. 13. U-net architecture for predicting dewpoint profiles. The initial RAP profile includes two variables—temperature and dewpoint—and is input to the beginning of the U-net (i.e., the first D-block layer). GOES data are included at the bottleneck layer, where they are concatenated with features extracted from RAP profiles by the D block. The output is the best-guess dewpoint profile, combining RAP and GOES data. Uncertainty is estimated at each layer in this profile. Adapted from Stock (2021), which did not include uncertainty estimates.

produce heavy rain, high winds, large hail, and tornadoes; all these phenomena threaten both lives and property. Currently, human forecasters rely on observations from radiosonde launches, which are spatially and temporally sparse, and simulations from numerical weather prediction (NWP) models. Stock (2021) used ML techniques to improve dewpoint vertical profiles, combining information from NWP models and satellite data, which are spatially and temporally denser than radiosonde data. Here we extend that work by adding UQ.

Specifically, we use a one-dimensional U-net architecture, adapted from Stock (2021), to predict dewpoint at 256 vertical levels. The predictors are a first-guess dewpoint profile from the Rapid Refresh NWP model (RAP; Benjamin et al. 2016), a temperature profile from the RAP, and satellite data from the *Geostationary Operational Environmental Satellite-16* (*GOES-16*) Advanced Baseline Imager (Schmit et al. 2017). The targets (ground truth) are dewpoint profiles from radiosonde observations (raob) over the central United States between 1 January 2017 and 31 August 2020. We use 75% of the data for training, 10% for validation, and 15% for testing. The model and experimental setup are described fully in Stock (2021); the model structure and the UQ methods we applied are shown in Fig. 13.

We train U-net models with four UQ methods (purple box in Fig. 13). The first two use PDP to predict normal (PDP_NORM) and SHASH (PDP_SHASH) distributions; the third uses EP with the CRPS loss function and 60 ensemble members (EP_CRPS); the last uses MC dropout with the MSE loss function (MC_DROPS), 60 ensemble members, and a dropout rate of 0.1 for all D blocks and U blocks. In earlier work (not shown) we experimented with varying the dropout rate from 0.01 to 0.5, as well as including versus not including dropout in the D blocks. These variations had minimal impact on the results shown.

b. *UQ results and discussion*

1) CASE STUDIES

Figure 14 shows three case studies for the PDP_NORM model, including both mean predictions and uncertainty estimates. As shown in Figs. 14a–c, the PDP_NORM model reasonably captures the y_{true} spread. Specifically, the 95% confidence interval contains the observed value y_{true} about 95% of the time. Also, the model's uncertainty usually increases with error in the mean prediction, suggesting that uncertainty and error are highly correlated (thus, we expect results of the discard test to be favorable). Beyond individual case studies, we have plotted a composite (Fig. 14d) by averaging over all the testing samples. Here we note that 1) the mean y_{true} and y_{pred} profiles are almost exactly the same, indicating that there is almost zero bias in the PDP_NORM model's mean predictions; 2) model uncertainty is lowest near the surface and highest in the mid–upper troposphere, around 300–500 hPa.

2) UQ-EVALUATION GRAPHICS

Evaluation graphics for all four UQ methods are shown in Fig. 15. Starting with the mean predictions, the attributes diagram (Fig. 15a) shows that each U-net model—regardless of which UQ method it uses—has well calibrated mean predictions, with the curve nearly following the 1-to-1 line. All models have an RMSE of ~5°C, and MSESS of 0.98, for the mean predictions. The largest conditional bias is a slight negative bias (~−2°C) for the lowest dewpoint predictions by PDP_SHASH. This bias likely occurs because the lowest dewpoints are underrepresented in the training data (i.e., rare events), as shown in the inset histogram. The remainder of this subsection focuses on evaluating uncertainty estimates, instead of the mean predictions.

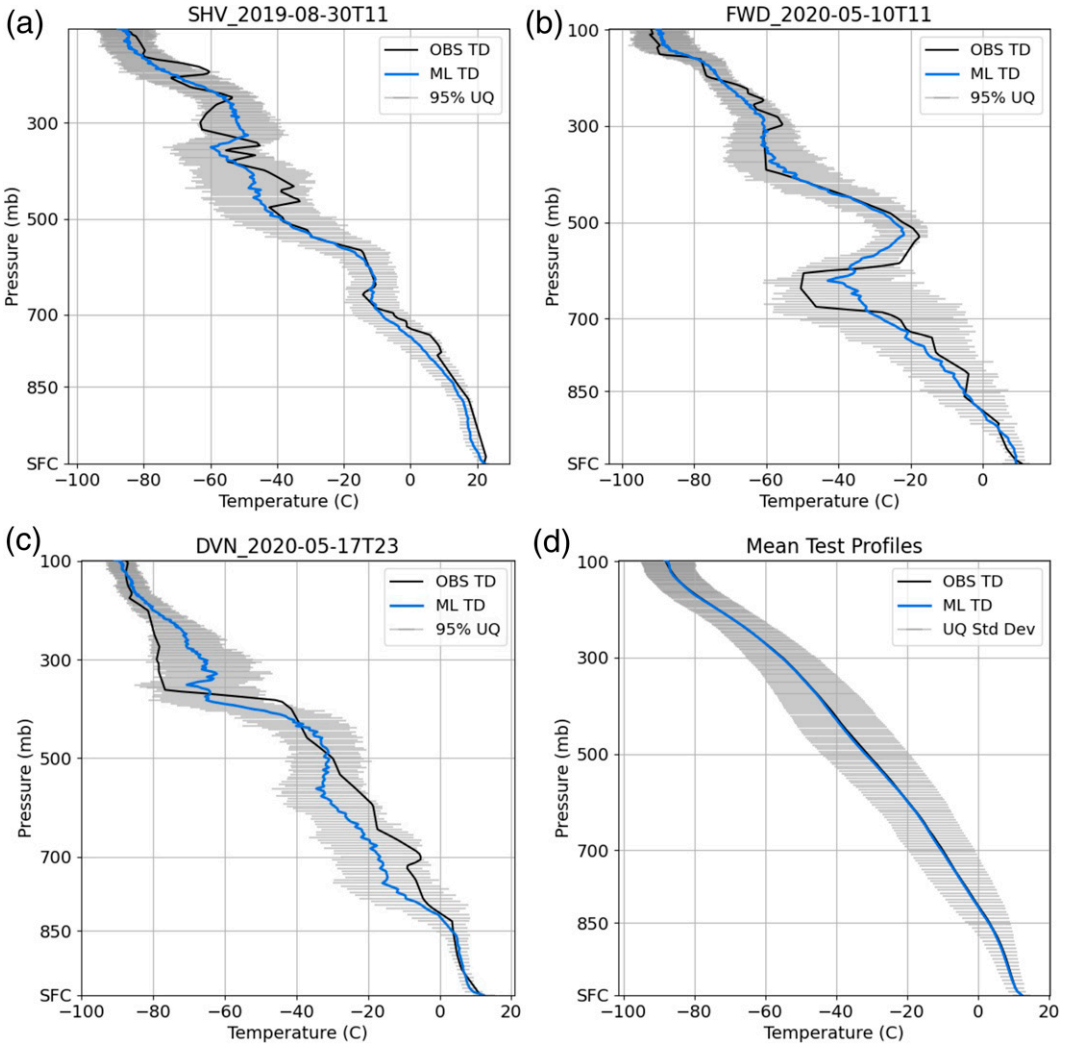


FIG. 14. Case studies for PDP_NORM model for predicting dewpoint profiles. In each panel, the observed profile y_{true} is shown in black, while the mean prediction and uncertainty estimate (properties of the y_{pred} distribution) are shown in blue and gray, respectively. (a)–(c) Individual case studies. Here, the uncertainty estimate shown is the 95% confidence interval. (d) Composite over all testing samples. Here, the uncertainty estimate shown is the mean of the predicted standard deviations.

The spread–skill plot (Fig. 15b) shows that for the vast majority of data points (those with y_{pred} spread below ~8°C), the PDP and EP approaches produce well calibrated uncertainty estimates, with the curve nearly following the 1-to-1 line. For the few data points with y_{pred} spread > 8°C, the PDP and EP approaches are underconfident; EP is the most underconfident. However, because EP samples directly from the y_{pred} distribution—rather than creating the y_{pred} distribution from just a few estimated parameters—EP can produce more complicated distributions. For example, EP can capture rare events where the y_{true} distribution is likely to resemble an extreme-value distribution instead of the typical canonical distributions assumed by PDP.

Although the flexibility of the EP approach is an advantage, it can complicate UQ evaluation, because for highly nonnormal distributions the standard deviation is a misleading measure of spread. Instead of using the standard deviation to quantify y_{pred} spread, it might be more appropriate to use the full histogram or PDF of the ensemble members. Allen et al. (2022) introduced the weighted PIT histogram, which is useful for rare events because it allows for regime-specific UQ evaluation (i.e., assessing forecast calibration as a function of the observed value). However, it is unclear how best to incorporate a histogram or PDF into an evaluation method like the spread–skill plot, so we leave this suggestion for future work.

Meanwhile, the MC-dropout approach is very overconfident and rarely produces y_{pred} spread above ~3°C (Fig. 15b). This result is not surprising, since MC dropout with a deterministic loss function (here, the MSE) cannot capture ML-aleatory uncertainty. This highlights the need to use a probabilistic loss function with MC dropout.

The discard test (Fig. 15c) shows that, for all four UQ methods, error (RMSE of the mean prediction) decreases

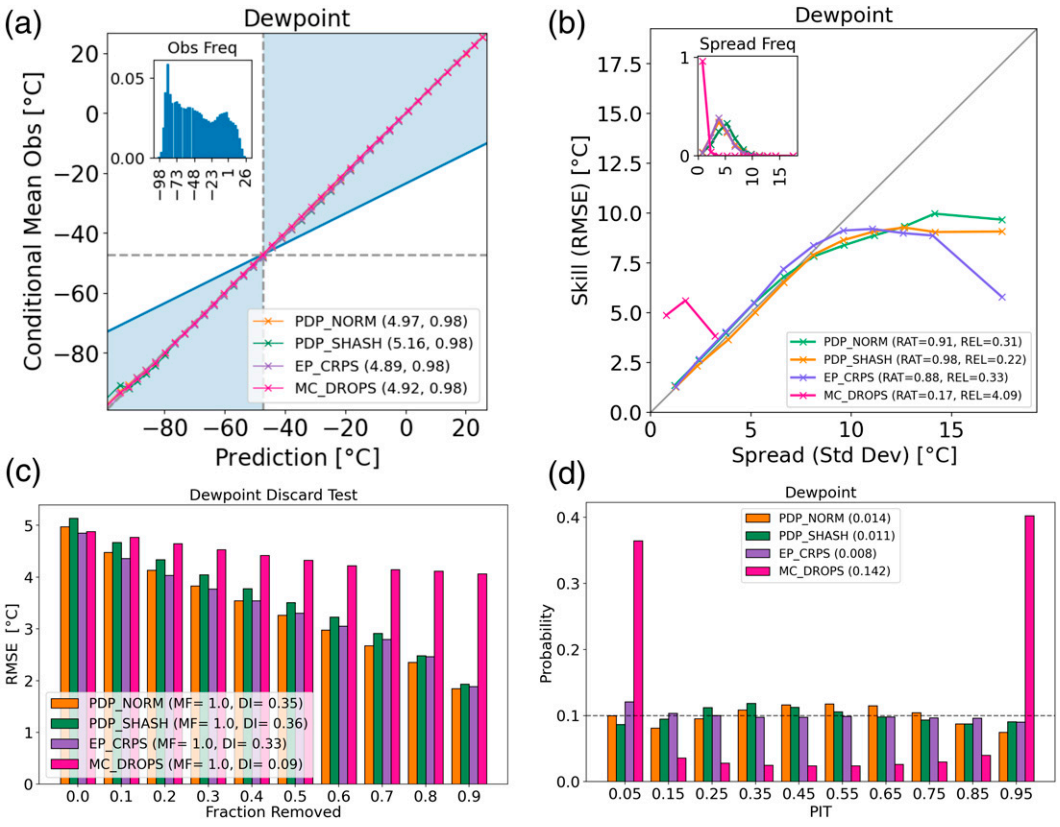


FIG. 15. Evaluation graphics for dewpoint prediction, showing U-nets trained with four different UQ methods: two PDP approaches (PDP_NORM and PDP_SHASH), an EP approach (EP_CRPS), and an MC-dropout approach (MC_DROPS). (a) Attributes diagram. The legend shows (RMSE, MSESS) for each model. (b) Spread–skill plot, including the SSRAT and SSREL for each model. (c) Discard test, including the MF and DI for each model. (d) PIT histogram, including the PITD for each model.

whenever discard fraction is increased—that is, whenever high-uncertainty cases are removed. In other words, all four models have an MF of 1.0. However, error decreases more sharply with discard fraction for the PDP and EP approaches (leading to DI $\geq$ 0.33) than for MC dropout (leading to DI = 0.09). These results highlight the importance of using both the MF and DI to summarize results of the discard test.

Finally, the PIT histogram (Fig. 15d) shows that the PDP and EP approaches produce well calibrated uncertainty estimates, with a nearly flat histogram and low PITD score. The PDP models have a slight hump in the middle of the histogram, indicating slight underconfidence (consistent with the spread–skill plot in Fig. 15b), while the EP model has a slight hump at the left edge of the histogram, indicating a slight overprediction bias (which cannot be seen in any of the other graphics in Fig. 15). Meanwhile, the MC-dropout model has large humps at both edges of the histogram, indicating that it is very overconfident (consistent with the spread–skill plot).

3) UQ-EVALUATION SCORES

Table 3 shows all eight scores discussed in section 4, plus the RMSE, for all four UQ methods on the dewpoint task. Qualitatively, results for this real-world task are nearly identical to results on the synthetic dataset (Table 2). Specifically, 1) mean predictions from the four models have nearly equal skill, with similar RMSE and MSESS values; 2) MC dropout produces the worst UQ estimates, as expected with a deterministic loss function; and 3) uncertainty estimates from the PDP and EP approaches have nearly equal skill, although the EP_CRPS model achieves the best score (6 times) more often than the PDP_SHASH model (5 times).

TABLE 3. Evaluation scores for the dewpoint task, with the best value for each score highlighted in bold.

Score	PDP_NORM	PDP_SHASH	EP_CRPS	MC_DROPS
RMSE	4.97	5.16	**4.89**	4.92
MSESS	**0.98**	**0.98**	**0.98**	**0.98**
SSRAT	0.91	**0.98**	0.88	0.17
SSREL	0.31	**0.22**	0.33	4.09
MF	**1.00**	**1.00**	**1.00**	**1.00**
DI	0.35	**0.36**	0.33	0.09
PITD	0.014	0.011	**0.008**	0.142
CRPS	2.52	2.62	**2.38**	3.07
IGN	4.36	4.40	**4.15**	8.81

6. Demonstration of UQ and evaluation methods for a classification task

In this section we introduce the classification task (predicting convection), apply MC dropout and quantile regression to obtain uncertainty estimates, then evaluate the uncertainty estimates and show case studies.

a. Predicting convection from satellite imagery

We adapt the work of Lagerquist et al. (2021, herein L21) to include uncertainty. The task is to predict the occurrence of thunderstorms (henceforth, "convection") at 1-h lead time at each pixel in a grid. As predictors, we use gridded brightness temperatures from the *Himawari-8* satellite at seven wavelengths and three lag times (0, 20, and 40 min before the forecast issue time t_0), as shown in Fig. 16. The target is a binary convection mask at $t_0 + 1$ h, created by applying an algorithm called "storm-labeling in 3 dimensions" (SL3D; Starzec et al. 2017) to radar data. Both the predictor and target variables are on a latitude–longitude grid with 0.0125° (~1.25 km) spacing. We train and evaluate the NNs only at pixels within 100 km of the nearest radar (gray circles in Figs. 16j,k), where radar coverage is sufficient to detect convection. See L21 for complete details.

Our NN architecture is a U-net specially designed to predict gridded variables—here, the presence of convection at $t_0 + 1$ h. Details on our particular architecture are in L21. The U-net in L21 is deterministic; in this paper we use either MC dropout or QR to make the U-net probabilistic. For MC dropout, we use the architecture in Fig. 17a. For QR, we replace the single output layer in Fig. 17a with $N + 1$ output layers, where N is the number of quantile levels estimated (Fig. 17b). We use Eq. (2) to prevent quantile crossing, as represented schematically in the rightmost column of Fig. 17b.[13] Hyperparameters not shown in Fig. 17 include batch normalization (we perform batch normalization after each ReLU activation) and the training procedure (we train for 1000 epochs with the Adam optimizer, an initial learning rate of 0.001, early stopping if validation loss has not improved over 30 epochs, and a 40% reduction in learning rate if validation loss has not improved over 10 epochs). These decisions are justified in Table S1 of L21.

Because convection is a rare event (occurring at only 0.75% of pixels), we need to use an aggressive loss function (one that rewards true positives more than true negatives). Traditional loss functions, like the traditional quantile loss [Eq. (1)], which reward true positives and true negatives equally, would result in a model that almost never predicts convection. Thus, we use a hybrid between the fractions skill score and traditional quantile loss, which we call the "aggressive quantile loss." See the appendix for details.

[13] For a regression problem, Eq. (2) alone is sufficient. For a classification problem like this one, quantile-based estimates $\hat{y}_i$ must range from [0, 1], allowing them to be interpreted as probabilities. This is why, as shown in Fig. 17b, we apply the sigmoid activation function to the output of Eq. (2).

b. UQ results and discussion

For MC dropout we tune three hyperparameters, specifically dropout rates for the last three layers (Table 4). For QR we tune two hyperparameters: the set of quantile levels and the weight w in Eq. (A2), which affects the importance of deterministic versus probabilistic predictions in the loss function (Table 4). Details are in section 1 of the online supplemental material. We run the MC-dropout models 100 times in inference mode, yielding an ensemble size of 100. We use the validation data to select hyperparameters and the independent testing data to show results for the selected models.

MONTE CARLO DROPOUT

As discussed in the online supplemental material, as dropout rates increase, the uncertainty estimates improve but the mean predictions deteriorate. Hence, there is a trade-off between the quality of probabilistic and deterministic predictions. We believe that this trade-off exists for two reasons. First, MC dropout leads to overconfident models,[14] so the easiest way for an MC-dropout model to improve uncertainty estimates is to produce higher spread. However, second, since MC dropout is a post hoc method not optimized to produce good uncertainty estimates, higher spread typically means lower skill (worse mean predictions). In our judgement (based on subjectively combining evaluation scores shown in the online supplemental material), the best model has a dropout rate of 0.250 for the last layer, 0.125 for the second-to-last layer, and 0.375 for the third-to-last layer.

Figure 18a shows the spread–skill plot for this model. The model is overconfident (underspread) for all bins; this problem is typical for MC dropout, including atmospheric-science applications (Scher and Messori 2021; Clare et al. 2021; Garg et al. 2022). Also, the model rarely produces spread values > 0.04, as shown in the histogram in Fig. 18a. The skewed histogram is explained by the inset in Fig. 18a: model spread increases with convection frequency (the mean target value), and since convection is a rare event, we should therefore expect high model spread to be a rare event. Figure 18b shows the discard test for the best model. Model error decreases almost every time the discard fraction is increased (18 of 19 times), yielding an MF of 94.74%. Thus, the ranking quality of the model's uncertainty estimates is high. As shown in the inset of Fig. 18b, event frequency decreases from 0.8% for all data samples to 0.2% for those not including the 10% with highest uncertainty. In other words, most convection is associated with very high uncertainty, consistent with the inset of Fig. 18a.

Figure 19 shows a case study created by applying the best model to one time step in the testing data, during Tropical Depression Luis. Figure 19 summarizes the predicted distribution with five numbers: the mean, standard deviation, and three percentiles of convection probability. We do not show percentiles below the 50th, because estimates corresponding to these

[14] By examining the spread–skill plots for all 125 models, we confirmed that they are all overconfident.

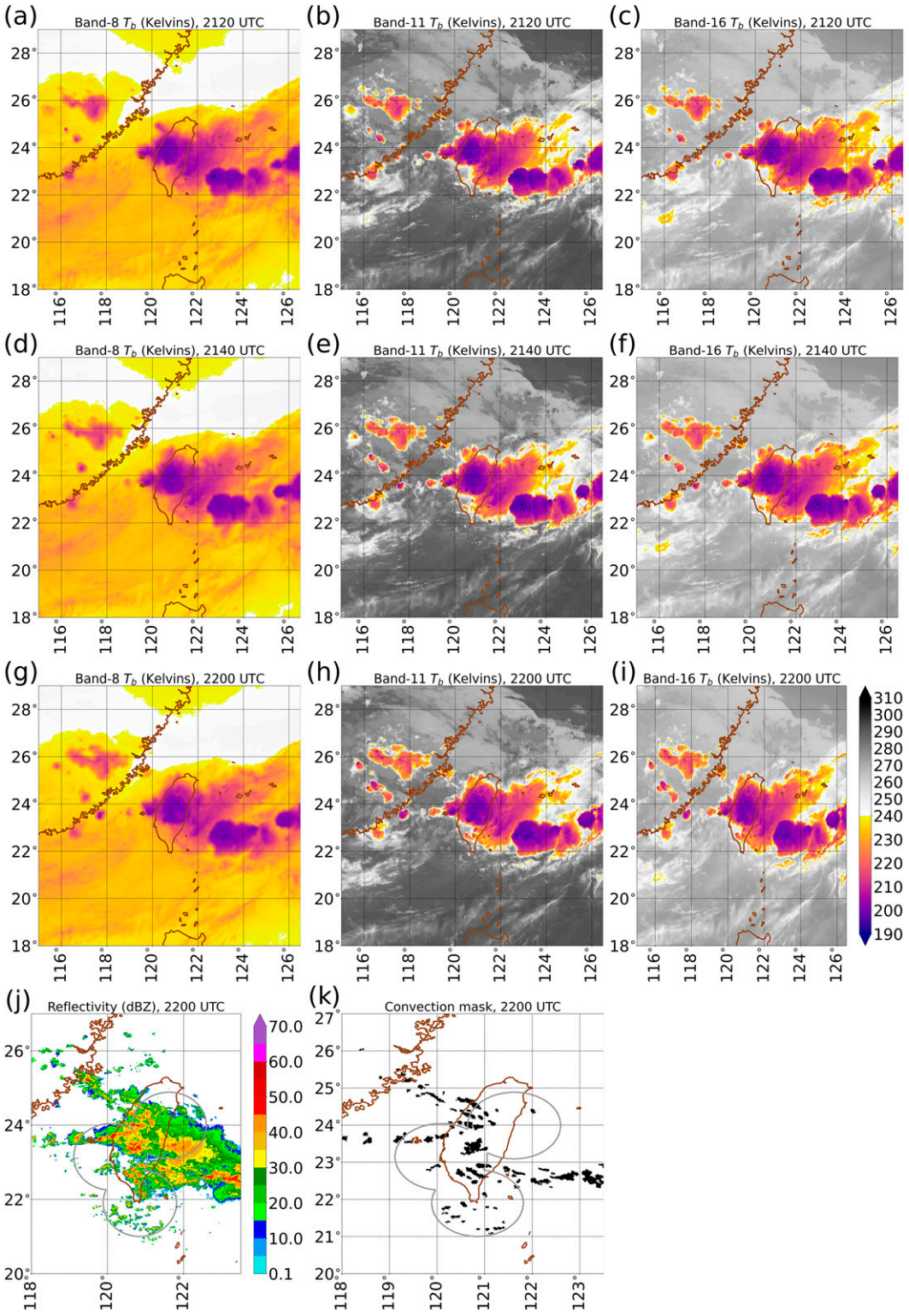


FIG. 16. Input data for predicting convection at 2200 UTC 2 Jun 2017. Shown are three of the seven predictors: band 8 (6.25 μm), band 11 (8.6 μm), and band 16 (13.3 μm). (a)–(c) Predictors at a lag time of 40 min; (d)–(f) predictors at a lag time of 20 min; (g)–(i) predictors at a lag time of 0 min. All predictors use the color bar next to (i). (j) Composite (column maximum) radar reflectivity. (k) Convection mask, the target variable. The black dots are pixels with true convection. Gray circles in (j) and (k) show the 100-km range ring around each radar.

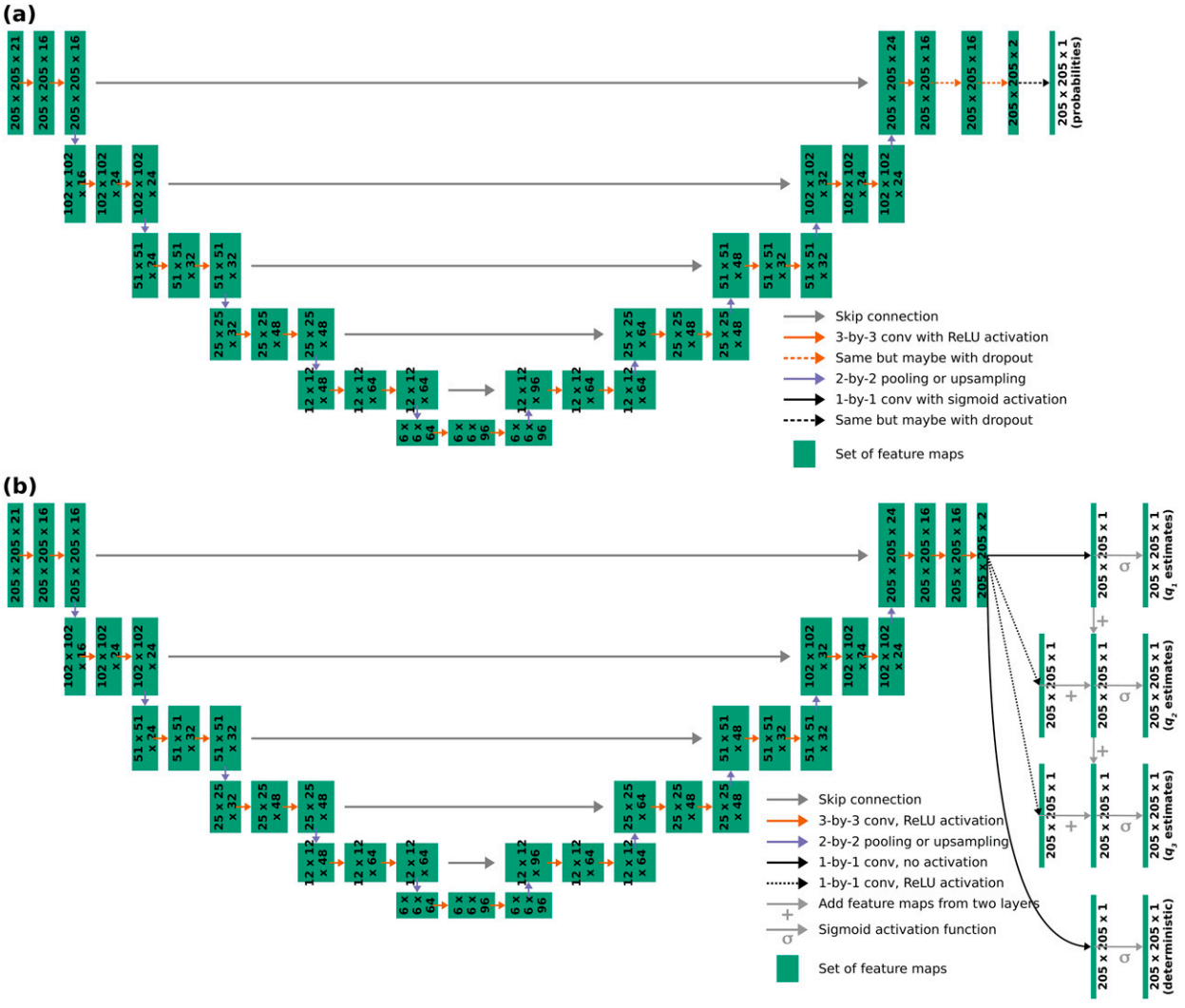


FIG. 17. U-net architecture for predicting convection from satellite imagery, using either (a) MC dropout or (b) quantile regression to estimate uncertainty. In each set of feature maps, the numbers are dimensions: $N_{\text{rows}} \times N_{\text{columns}} \times N_{\text{channels}}$. The 21 input channels are the raw predictor variables, i.e., gridded brightness temperatures at 7 wavelengths and 3 lag times. In (a), the last three convolutional layers are marked with dashed lines (either orange or black), indicating that these layers may include MC dropout. In (b), to be brief, we assume that there are only three quantile levels to estimate. When there are more quantile levels (which is the case for every U-net involved in this study), the pattern on the right side of (b) repeats. Specifically, estimates for quantile level q_i are created by applying 1×1 convolution with ReLU to the feature maps marked "$205 \times 205 \times 2$," adding the result to presigmoid estimates for quantile level q_{i-1}, then applying the sigmoid activation function. The sigmoid activation function constrains the final estimates to the range [0, 1], so that they may be interpreted as probabilities of convection.

percentiles are usually very small; most of the variation in the predicted distribution is between the 50th and 97.5th percentiles. The case shown in Fig. 19 features two large areas of convection: strong ("S") and weak ("W"). We make the following observations. First, in terms of the mean and any percentile, the model produces higher probabilities for strong convection than for weak convection (Figs. 19a,c–e), which is a desired property. Second, the model is more uncertain for strong convection than for weak convection (Fig. 19b). This is not a desired property, because weak convection (borderline cases) is more difficult to identify and should have higher uncertainty. Third, the probability maps contain checkerboard artifacts (Figs. 19a–c); these are caused by using dropout in the last layer, which sets some probabilities to zero. Fourth, the overconfidence (underspread) problem with MC dropout is obvious in area W, where there is almost no difference between the 50th and 97.5th percentile estimates (Figs. 19c–e).

c. *QR*

As discussed in the online supplemental material, results of the hyperparameter experiment are noisy. In our judgement (based on subjectively combining evaluation scores shown in

TABLE 4. Experimental hyperparameters for the classification task (nowcasting convection). Both hyperparameter experiments use the grid-search algorithm (section 11.4.3 of Goodfellow et al. 2016). Thus, we train 125 NNs with MC dropout (5 dropout rates for third-to-last layer × 5 rates for second-to-last layer × 5 rates for last layer) and 90 NNs with QR (9 sets of quantile levels × 10 weights. Exact quantile levels used (rather than just the number of quantiles) are shown in Table S1 in the online supplemental material.

Hyperparameter	Values attempted
MC-dropout experiment	
Dropout rate for third-to-last layer	0.000, 0.125, 0.250, 0.375, 0.500
Dropout rate for second-to-last layer	See above
Dropout rate for last layer	See above
QR experiment	
Number of quantile levels	17, 18, 19, 22, 26, 30, 38, 53, 101
Loss-function weight [Eq. (A2)]	1, 2, 3, 4, 5, 6, 7, 8, 9, 10

the online supplemental material) the best model has 17 quantile levels and a weight of 6 [$w = 6$ in Eq. (A2)]. To compute the model spread, defined as the standard deviation of the predicted distribution, we use Eq. (15) of Wan et al. (2014).

The spread–skill plot for the best model shows that it is almost perfectly calibrated when the y_{pred} distribution has spread ≤ 0.06 and underconfident (overspread) when y_{pred} spread > 0.06 (Fig. 18c). However, spread rarely exceeds 0.06 (only for 22.7% of examples, as shown by the histogram), so the model is generally well calibrated. The spread–skill plots reveal two advantages of the QR model (Fig. 18c) over the MC-dropout model (Fig. 18a). First, the QR model is better-calibrated overall, which manifests in a substantially lower SSREL (0.027 vs 0.037); the difference is significant at the 95% confidence level.[15] Second, the QR model's spread histogram is less skewed, that is, the QR model produces high spread values (>0.04) more often. However, the QR model is not much better-calibrated at these high spread values. As mentioned above, the overconfidence problem for MC dropout (which occurs for all 125 models we trained) is well documented in the literature. However, to our knowledge, the underconfidence problem for QR (which occurs for all 90 models we trained) has not been noted previously. Hence, this may be a problem with QR in general or specific to our application.

Figure 18d shows the discard test for the best QR model. The error decreases every time the discard fraction is increased, yielding an MF of 100%, compared to 94.74% for the MC-dropout model. Also, error decreases more sharply with discard fraction for the QR model than for the MC-dropout model (Fig. 18b), leading to a higher DI score (0.0033 vs 0.0030).

[15] Determined by a one-sided paired bootstrap test with 1000 iterations.

Figure 20 shows a case study for the best QR model, at the same time step as the MC-dropout case study (Fig. 19). We make the following observations. First, like the MC-dropout model, the QR model produces higher probabilities for stronger convection[16] (Figs. 20a,c,d). Second, according to the standard deviation (Fig. 20b), the QR model is more uncertain for the strong convection than the weak convection—a disadvantage shared by the MC-dropout model. The third observation counteracts the second: the standard deviation is not the full story on uncertainty, and sometimes it is necessary to look at the full predicted distribution. The MC-dropout model has almost no difference between the 50th and 97.5th percentile estimates in area W (Figs. 19c–e), consistent with the low standard deviations (Fig. 19b). However, the QR model has large differences between the 50th and 97.5th percentile estimates in area W (Figs. 20c–e), despite the standard deviations here being smaller than elsewhere in the domain (Fig. 20b). Fourth, the overall underconfidence of the QR model—shown in Fig. 18c—is manifested in the case study, where the 97.5th percentile estimate is essentially 100% everywhere in the domain (Fig. 20e). For an additional case study during the winter, see section 2 of the online supplemental material.

This section details many advantages of the QR model over the MC-dropout model. However, note that the MC-dropout model has a substantially lower CRPS (0.020 vs 0.034). This highlights that single-number summaries are not the full story and should be accompanied by a detailed investigation, including standard evaluation graphics and case studies.

7. Discussion and conclusions

Uncertainty quantification (UQ) is a key tool for understanding ML models. For applications that involve critical decision-making, UQ is invaluable for assessing the trustworthiness of the model. Recent years have seen a surge in research on UQ methods for ML, especially neural networks (NN). However, as more UQ methods are developed and applied in domains such as environmental science, it is essential for domain scientists to understand the UQ approaches and how to evaluate them—that is, how to determine whether the uncertainty estimates are good. If uncertainty estimates are poor, they can easily increase trust in an ML model without increasing the model's actual *trustworthiness*.

To this end, we have summarized six popular UQ approaches, four UQ-evaluation graphics, and eight UQ-evaluation scores (single-number summaries)—with the goal of making all these tools accessible to the environmental-science community. We have included sample Python code implementing most of the UQ approaches, and all of the UQ-evaluation methods, for NNs. Last, we have applied several UQ approaches and UQ-evaluation methods to two real-world applications in atmospheric science: predicting dewpoint profiles (a regression task) and nowcasting the locations of convection (a classification task).

[16] However, at the 97.5th percentile of the predicted distribution, the convection probability is ~100% at all pixels (Fig. 20e), regardless of whether the pixel contains strong or weak or no convection.

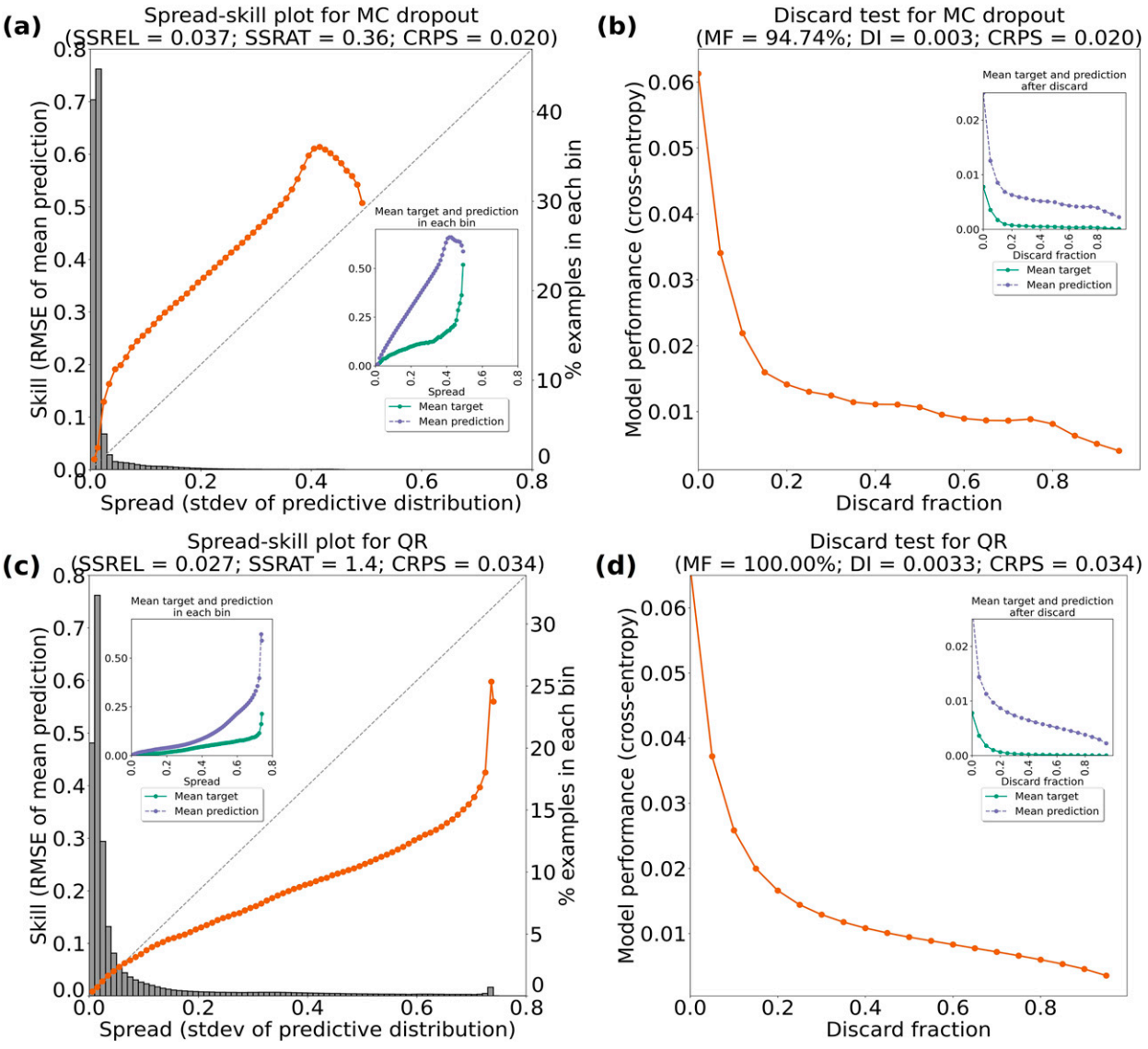


FIG. 18. (a) Spread–skill plot and (b) discard test, measured on testing data, when using MC dropout to predict convection. (c) Spread–skill plot and (d) discard test, measured on testing data, when using quantile regression to predict convection. The histogram in the spread–skill plot shows the percentage of testing examples in each bin of spread values; the bins have a spacing of 0.01.

To summarize our findings on UQ-evaluation methods:

- The attributes diagram provides an excellent way to evaluate the central predictions, revealing where the predictions are performing poorly and indicating biases.
- The spread–skill plot comprehensively describes how the model error compares to the model uncertainty prediction, indicating 1) if the uncertainty estimates are well calibrated or conditionally biased (i.e., under- or overconfident); 2) how often the model performs well (or poorly).
- Both the spread–skill plot and PIT histogram display uncertainty calibration. The spread–skill plot can identify under/overconfidence as a function of model spread, while the PIT histogram cannot. The PIT histogram can identify under/overprediction (whether y_{true} falls too often near the top/bottom of the y_{pred} distribution, respectively), while the spread–skill plot cannot.
- The discard test shows whether error is correlated with the uncertainty, quickly identifying if model performance improves by removing the cases with the highest uncertainty. In an operational setting, information conveyed by the discard test can be very useful. For example, if users have a threshold for maximum acceptable error (which is not available in real time), they can set a corresponding threshold on model spread (which is available in real time).
- The evaluation scores provide a quick way to compare results between models; however, the graphics allow for deeper insight. We suggest using them together, because they provide complementary information. For example, in our real-world convection application, the often-used CRPS

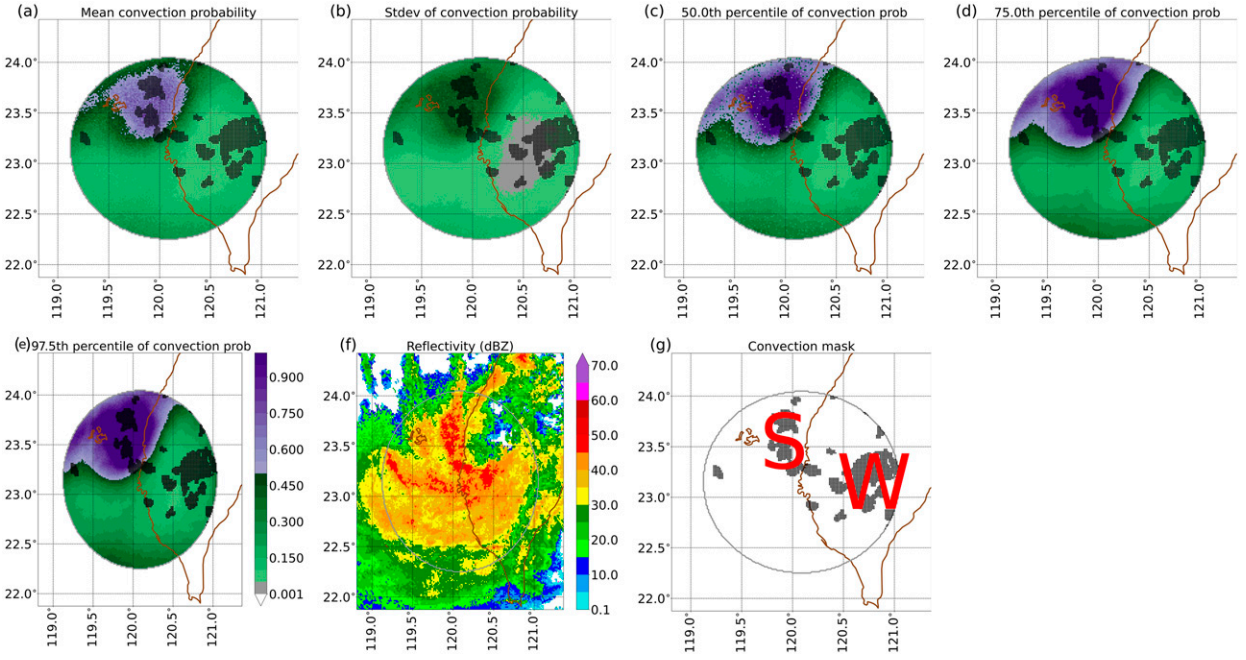


FIG. 19. Case study for the MC-dropout model, during Tropical Depression Luis. All data (predictions, radar reflectivity, and convection mask) are valid at 0830 UTC 23 Aug 2018. The predictions were made with 1-h lead time (initialized at 0730 UTC). (a) Mean convection probability; (b) standard deviation of convective probability; (c) median convection probability; (d) 75th percentile convection probability; (e) 97.5th percentile convection probability; (f) composite (column-maximum) radar reflectivity; and (g) true convection mask, with black dots showing convective pixels. "S" indicates an area of strong convection west of the center of Luis, while "W" indicates an area of weak convection east of the center.

suggested that MC dropout was a better UQ method than quantile regression, but all graphics suggested the opposite.
- We caution that evaluation methods using a single number to quantify model spread—like the standard deviation, which is always used in the spread–skill plot and typically used for the discard test—may be misleading for highly nonnormal distributions. UQ-evaluation methods could possibly be altered to include more holistic information, such as the full histogram or PDF, but we leave these suggestions for future work.

Case studies are useful not only to understand a model's behavior in a way that cannot be conveyed by averaging over a large dataset, but also to examine the shape of the y_{pred} distribution.

Applying the UQ approaches to real-world atmospheric-science problems yielded the following findings.

- The PDP, NPDP, and EP approaches perform well on average but are underconfident for high-error cases.
- PDP and EP perform very similarly for the regression task, producing well calibrated uncertainty estimates.
- The PDP approach with a normal distribution is the easiest to implement, and it provided similar enough estimates (even with asymmetric spread in y_{true}) to other approaches to justify its use.
- For applications with complicated y_{true} spread (e.g., highly nonnormal distributions or predicting rare/extreme events), EP is a better approach a priori, because it is more flexible and can theoretically match any type of distribution.
- QR, although typically used for regression tasks, can be useful for classification tasks. Compared to MC dropout on the convection application, QR produces a better spread–skill plot, a better discard test, and more useful uncertainty estimates in the case studies shown. However, for all 90 models, each with different hyperparameters, we noted that QR is underconfident. To our knowledge this underconfidence problem is not documented in the literature, so it could be a general problem with QR or one specific to our convection application.
- MC dropout is consistently the worst-performing approach, providing overconfident uncertainty estimates. This is a well-known problem in the literature; it arises because, when trained with a deterministic loss function (the standard approach), MC dropout only captures ML-epistemic uncertainty. However, due to its ease of implementation, we believe that MC dropout should still be used as a baseline against which to compare more sophisticated UQ methods. If trained with a probabilistic loss function, MC dropout can theoretically capture ML-aleatory uncertainty as well.

Uncertainty estimation is a prevalent topic across a broad range of communities; however, unfortunately the definitions of aleatory and epistemic uncertainty are inconsistent between disciplines, so as we incorporate UQ into environmental-science applications, we need to be exact in our use of these terms. This work focuses on capturing ML-aleatory uncertainty, which is directly calculable from the data. However, total

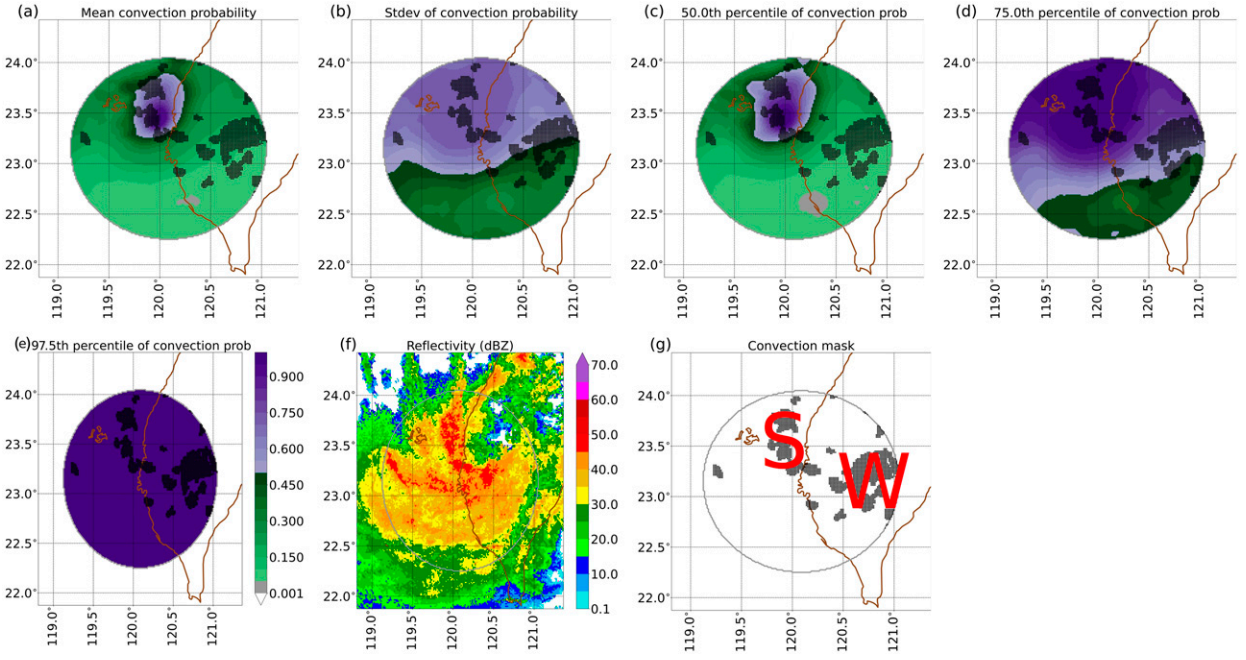


FIG. 20. As in Fig. 19, but for the QR model.

uncertainty also includes the ML-epistemic component, and the best way to calculate this is still a topic of active research. For example, how does one choose testing data with enough out-of-regime samples to robustly estimate ML-epistemic uncertainty? In future work we will implement BNNs, which can capture both types of uncertainty when combined with another UQ approach. With the recent surge in UQ approaches and evaluation techniques, the challenge is shifting from UQ implementation and evaluation to ensuring their proper use, including how to effectively interpret and communicate the information that they provide.

Acknowledgments. This work was funded in part by NSF AI Institute Grant ICER-2019758 (MM, IE), NSF HDR Grant OAC-1934668 (IE), and NOAA Grant NA19OAR4320073 (all authors). We thank Dr. Steven Brey for providing his expertise on the CRPS method and making his CRPS-Net code available.

Data availability statement. The Jupyter notebooks referenced throughout this article are available in a GitHub repository, which you can find at https://github.com/thunderhoser/cira_uq4ml. Additionally, we have created a frozen release of the notebooks for this article at https://doi.org/10.5281/zenodo.7335705.

APPENDIX

Aggressive Quantile Loss for Predicting Convection

The fractions skill score (FSS; Roberts and Lean 2008) rewards true positives more than true negatives, making it well suited for rare-event prediction. The U-net for QR (Fig. 17b) has $N + 1$ outputs, where N is the number of quantile levels estimated. The last output is the deterministic prediction, and its loss function is the pixelwise FSS (i.e., FSS without a neighborhood filter). The ith output ($i \leq N$) is the estimate for quantile level q_i, and its loss function is a hybrid between the traditional quantile loss [Eq. (1)] and the pixelwise FSS:

$$\mathscr{L} = \begin{cases} (1 - q_i)\dfrac{(y_{\text{true}} - y_{\text{pred}}^{q_i})^2}{y_{\text{true}}^2 + y_{\text{pred}}^{q_i\,2}}, & y_{\text{true}} \leq y_{\text{pred}}^{q_i}; \\ q_i \dfrac{(y_{\text{true}} - y_{\text{pred}}^{q_i})^2}{y_{\text{true}}^2 + y_{\text{pred}}^{q_i\,2}}, & y_{\text{true}} > y_{\text{pred}}^{q_i}. \end{cases} \tag{A1}$$

The term y_{true} is the true value from the convection mask, and $y_{\text{pred}}^{q_i} \in [0, 1]$ is the estimate for quantile level q_i. You might ask: why use the FSS without a neighborhood filter? After all, a key benefit of the FSS is that it uses a neighborhood filter to solve the double-penalty problem, where a model is punished too harshly for a small (e.g., 1 pixel) offset between the predicted and observed event. The answer is that 1) convection is a rare event, occurring at only 0.75% of pixels on average; 2) to obtain a U-net that predicts substantial probabilities for a rare event, it is necessary to use a loss function that rewards true positives more than true negatives; 3) the FSS, even without a neighborhood filter, has this desired property; 4) including a filter in the loss function requires the inclusion of a filter in the UQ evaluation metrics, which is more complication than we wanted. We call the loss function in Eq. (A1) the "aggressive quantile loss."

To compute the total loss for the ith output ($i \leq N$), we average Eq. (A1) over all pixels and valid times, yielding $\overline{\mathscr{L}_i}$. To compute the total loss for the U-net, we use the equation

$$\overline{\mathscr{L}_{\text{model}}} = w\overline{\mathscr{L}_{\text{deterministic}}} + \frac{1}{N}\sum_{i=1}^{N}\overline{\mathscr{L}_i}, \tag{A2}$$

where $\overline{\mathscr{L}_{\text{deterministic}}}$ is the loss for the deterministic prediction (pixelwise FSS) and wge1 is a user-selected weight. When there are many quantile levels (i.e., N is large), this weight is needed to emphasize the deterministic predictions, ensuring that both deterministic and probabilistic predictions are skillful.

REFERENCES

Allen, S., J. Bhend, O. Martius, and J. Ziegel, 2022: Weighted verification tools to evaluate univariate and multivariate forecasts for high-impact weather events. arXiv, 2209.04872v1, https://doi.org/10.48550/arXiv.2209.04872.

Baran, S., and A. Baran, 2021: Calibration of wind speed ensemble forecasts for power generation. *Weather*, **125**, 609–624, https://doi.org/10.28974/idojaras.2021.4.4.

Barnes, E., and R. J. Barnes, 2021: Controlled abstention neural networks for identifying skillful predictions for regression problems. *J. Adv. Model. Earth Syst.*, **13**, e2021MS002575, https://doi.org/10.1029/2021MS002575.

——, ——, and N. Gordillo, 2021: Adding uncertainty to neural network regression tasks in the geosciences. arXiv, 2109.07250v1, https://doi.org/10.48550/arXiv.2109.07250.

Beck, J., F. Bouttier, L. Wiegand, C. Gebhardt, C. Eagle, and N. Roberts, 2016: Development and verification of two convection-allowing multi-model ensembles over Western Europe. *Quart. J. Roy. Meteor. Soc.*, **142**, 2808–2826, https://doi.org/10.1002/qj.2870.

Benjamin, S. G., and Coauthors, 2016: A North American hourly assimilation and model forecast cycle: The Rapid Refresh. *Mon. Wea. Rev.*, **144**, 1669–1694, https://doi.org/10.1175/MWR-D-15-0242.1.

Beucler, T., I. Ebert-Uphoff, S. Rasp, M. Pritchard, and P. Gentine, 2021: Machine learning for clouds and climate (invited chapter for the AGU Geophysical Monograph Series "Clouds and Climate"). ESS Open Archive, 28 pp., https://doi.org/10.1002/essoar.10506925.1.

Bevan, L. D., 2022: The ambiguities of uncertainty: A review of uncertainty frameworks relevant to the assessment of environmental change. *Futures*, **137**, 102919, https://doi.org/10.1016/j.futures.2022.102919.

Bihlo, A., 2021: A generative adversarial network approach to (ensemble) weather prediction. *Neural Networks*, **139**, 1–16, https://doi.org/10.1016/j.neunet.2021.02.003.

Bremnes, J., 2020: Ensemble postprocessing using quantile function regression based on neural networks and Bernstein polynomials. *Mon. Wea. Rev.*, **148**, 403–414, https://doi.org/10.1175/MWR-D-19-0227.1.

Brey, S., 2021: Ensemble. GitHub, https://github.com/TheClimateCorporation/ensemble.

Chapman, W. E., L. D. Monache, S. Alessandrini, A. C. Subramanian, F. M. Ralph, S.-P. Xie, S. Lerch, and N. Hayatbini, 2022: Probabilistic predictions from deterministic atmospheric river forecasts with deep learning. *Mon. Wea. Rev.*, **150**, 215–234, https://doi.org/10.1175/MWR-D-21-0106.1.

Clare, M. C. A., O. Jamil, and C. J. Morcrette, 2021: Combining distribution-based neural networks to predict weather forecast probabilities. *Quart. J. Roy. Meteor. Soc.*, **147**, 4337–4357, https://doi.org/10.1002/qj.4180.

Dai, Y., and S. Hemri, 2021: Spatially coherent postprocessing of cloud cover forecasts using generative adversarial networks. *EGU General Assembly* 2021, Online, European Geosciences Union, EGU21–4374, https://doi.org/10.5194/egusphere-egu21-4374.

Delle Monache, L., F. A. Eckel, D. L. Rife, B. Nagarajan, and K. Searight, 2013: Probabilistic weather prediction with an analog ensemble. *Mon. Wea. Rev.*, **141**, 3498–3516, https://doi.org/10.1175/MWR-D-12-00281.1.

DelSole, T., X. Yang, and M. K. Tippett, 2013: Is unequal weighting significantly better than equal weighting for multi-model forecasting? *Quart. J. Roy. Meteor. Soc.*, **139**, 176–183, https://doi.org/10.1002/qj.1961.

Demuth, J., and Coauthors, 2023: When will it start and when will it end? Developing ensemble-based guidance to predict timing for winter and fire weather based on user needs. *13th Conf. on Transition of Research to Operations*, Denver, CO, Amer. Meteor. Soc., 9B.6, https://ams.confex.com/ams/103ANNUAL/meetingapp.cgi/Paper/418286.

Doblas-Reyes, F., R. Hagedorn, and T. Palmer, 2005: The rationale behind the success of multi-model ensembles in seasonal forecasting–II. Calibration and combination. *Tellus*, **57A**, 234–252, https://doi.org/10.3402/tellusa.v57i3.14658.

Dürr, O., B. Sick, and E. Murina, 2020: *Probabilistic Deep Learning: With Python, Keras and Tensorflow Probability*. 1st ed. Manning Publications, 296 pp.

Gal, Y., and Z. Ghahramani, 2016: Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. *Proceedings of the 33rd International Conference on Machine Learning*, Vol. 48, Association for Computing Machinery, 1050–1059, https://dl.acm.org/doi/10.5555/3045390.3045502.

Garg, S., S. Rasp, and N. Thuerey, 2022: WeatherBench probability: A benchmark dataset for probabilistic medium-range weather forecasting along with deep learning baseline models. arXiv, 2205.00865v1, https://doi.org/10.48550/arXiv.2205.00865.

Ghazvinian, M., Y. Zhang, D.-J. Seo, M. He, and N. Fernando, 2021: A novel hybrid artificial neural network-parametric scheme for postprocessing medium-range precipitation forecasts. *Adv. Water Resour.*, **151**, 103907, https://doi.org/10.1016/j.advwatres.2021.103907.

Gneiting, T., and A. Raftery, 2007: Strictly proper scoring rules, prediction, and estimation. *J. Amer. Stat. Assoc.*, **102**, 359–378, https://doi.org/10.1198/016214506000001437.

——, A. E. Raftery, A. H. Westveld, and T. Goldman, 2005: Calibrated probabilistic forecasting using ensemble model output statistics and minimum CRPS estimation. *Mon. Wea. Rev.*, **133**, 1098–1118, https://doi.org/10.1175/MWR2904.1.

Good, I. J., 1952: Rational decisions. *J. Roy. Stat. Soc.*, **148B**, 107–114, https://doi.org/10.1111/j.2517-6161.1952.tb00104.x.

Goodfellow, I., Y. Bengio, and A. Courville, 2016: *Deep Learning*. MIT Press, 800 pp.

Hamill, T. M., 2001: Interpretation of rank histograms for verifying ensemble forecasts. *Mon. Wea. Rev.*, **129**, 550–560, https://doi.org/10.1175/1520-0493(2001)129<0550:IORHFV>2.0.CO;2.

Hersbach, H., 2000: Decomposition of the continuous ranked probability score for ensemble prediction systems. *Wea. Forecasting*,

15, 559–570, https://doi.org/10.1175/1520-0434(2000)015<0559:DOTCRP>2.0.CO;2.

Hinton, G., N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, 2012: Improving neural networks by preventing co-adaptation of feature detectors. arXiv, 1207.0580v1, https://doi.org/10.48550/arXiv.1207.0580.

Hsu, W.-r., and A. H. Murphy, 1986: The attributes diagram: A geometrical framework for assessing the quality of probability forecasts. *Int. J. Forecasting*, **2**, 285–293, https://doi.org/10.1016/0169-2070(86)90048-8.

Hüllermeier, E., and W. Waegeman, 2021: Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Mach. Learn.*, **110**, 457–506, https://doi.org/10.1007/s10994-021-05946-3.

Jospin, L. V., H. Laga, F. Boussaid, W. Buntine, and M. Bennamoun, 2022: Hands-on Bayesian neural networks—A tutorial for deep learning users. *IEEE Comput. Intell. Mag.*, **17**, 29–48, https://doi.org/10.1109/MCI.2022.3155327.

Klotz, D., F. Kratzert, M. Gauch, A. Keefe Sampson, J. Brandstetter, G. Klambauer, S. Hochreiter, and G. Nearing, 2022: Uncertainty estimation with deep learning for rainfall–runoff modelling. *Hydrol. Earth Syst. Sci.*, **26**, 1673–1693, https://doi.org/10.5194/hess-26-1673-2022.

Lagerquist, R., J. Q. Stewart, I. Ebert-Uphoff, and C. Kumler, 2021: Using deep learning to nowcast the spatial coverage of convection from *Himawari-8* satellite data. *Mon. Wea. Rev.*, **149**, 3897–3921, https://doi.org/10.1175/MWR-D-21-0096.1.

Matheson, J. E., and R. L. Winkler, 1976: Scoring rules for continuous probability distributions. *Manage. Sci.*, **22**, 1087–1096, https://doi.org/10.1287/mnsc.22.10.1087.

Meinshausen, N., and G. Ridgeway, 2006: Quantile regression forests. *J. Mach. Learn. Res.*, **7**, 983–999.

Nair, V., and G. Hinton, 2010: Rectified linear units improve restricted Boltzmann machines. *Proc. 27th Int. Conf. on Machine Learning*, Haifa, Israel, Association for Computing Machinery, 807–814, https://dl.acm.org/doi/10.5555/3104322.3104425

Nipen, T., and R. Stull, 2011: Calibrating probabilistic forecasts from an NWP ensemble. *Tellus*, **63A**, 858–875, https://doi.org/10.1111/j.1600-0870.2011.00535.x.

Orescanin, M., V. Petković, S. W. Powell, B. R. Marsh, and S. C. Heslin, 2021: Bayesian deep learning for passive microwave precipitation type detection. *IEEE Geosci. Remote Sens. Lett.*, **19**, 4500705, https://doi.org/10.1109/LGRS.2021.3090743.

Ortiz, P., M. Orescanin, V. Petković, S. W. Powell, and B. Marsh, 2022: Decomposing satellite-based classification uncertainties in large earth science datasets. *IEEE Trans. Geosci. Remote Sens.*, **60**, 1–11, https://doi.org/10.1109/TGRS.2022.3152516.

Rasp, S., and S. Lerch, 2018: Neural networks for postprocessing ensemble weather forecasts. *Mon. Wea. Rev.*, **146**, 3885–3900, https://doi.org/10.1175/MWR-D-18-0187.1.

Roberts, N. M., and H. W. Lean, 2008: Scale-selective verification of rainfall accumulations from high-resolution forecasts of convective events. *Mon. Wea. Rev.*, **136**, 78–97, https://doi.org/10.1175/2007MWR2123.1.

Rogers, P., K. Serr, K. Deitsch, J. Demuth, and P. Schumacher, 2023: How NWS partners use and understand snowfall probabilities: Part I—Survey overview. *13th Conf. on Transition of Research to Operations*, Denver, CO, Amer. Meteor. Soc., 12A.4, https://ams.confex.com/ams/103ANNUAL/meetingapp.cgi/Paper/412957.

Roulston, M. S., and L. A. Smith, 2002: Evaluating probabilistic forecasts using information theory. *Mon. Wea. Rev.*, **130**, 1653–1660, https://doi.org/10.1175/1520-0493(2002)130<1653:EPFUIT>2.0.CO;2.

Salama, K., 2021: Probabilistic Bayesian neural networks. Keras, https://keras.io/examples/keras_recipes/bayesian_neural_networks/.

Sato, S., M. Takanashi, K. Indo, N. Nishihara, H. Ichikawa, and H. Watanabe, 2021: Two-stage probabilistic short-term wind power prediction using neural network with MC dropout and control information. *Int. Exhibition and Conf. for Power Electronics, Intelligent Motion, Renewable Energy and Energy Management*, Online, Institute of Electrical and Electronics Engineers, 1–8, https://ieeexplore.ieee.org/abstract/document/9472407.

Scher, S., and G. Messori, 2021: Ensemble methods for neural network-based weather forecasts. *J. Adv. Model. Earth Syst.*, **13**, e2020MS002331, https://doi.org/10.1029/2020MS002331.

Scheuerer, M., M. B. Switanek, R. P. Worsnop, and T. M. Hamill, 2020: Using artificial neural networks for generating probabilistic subseasonal precipitation forecasts over California. *Mon. Wea. Rev.*, **148**, 3489–3506, https://doi.org/10.1175/MWR-D-20-0096.1.

Schmit, T. J., P. Griffith, M. M. Gunshor, J. M. Daniels, S. J. Goodman, and W. J. Lebair, 2017: A closer look at the ABI on the GOES-R series. *Bull. Amer. Meteor. Soc.*, **98**, 681–698, https://doi.org/10.1175/BAMS-D-15-00230.1.

Schulz, B., and S. Lerch, 2022: Machine learning methods for postprocessing ensemble forecasts of wind gusts: A systematic comparison. *Mon. Wea. Rev.*, **150**, 235–257, https://doi.org/10.1175/MWR-D-21-0150.1.

Serr, K., P. Rogers, K. Deitsch, J. Demuth, P. Schumacher, R. Prestley, and C. Wirz, 2023: How NWS partners use and understand snowfall probabilities: Part II—Survey results. *13th Conf. on Transition of Research to Operations*, Denver, CO, Amer. Meteor. Soc., 12A.5, https://ams.confex.com/ams/103ANNUAL/meetingapp.cgi/Paper/412965.

Starzec, M., C. R. Hometer, and G. L. Mullendore, 2017: Storm labeling in three dimensions (SL3D): A volumetric radar echo and dual-polarization updraft classification algorithm. *Mon. Wea. Rev.*, **145**, 1127–1145, https://doi.org/10.1175/MWR-D-16-0089.1.

Stock, J. D., 2021: Using machine learning to improve vertical profiles of temperature and moisture for severe weather nowcasting. M.S. thesis, Dept. of Computer Science, Colorado State University, 95 pp., https://hdl.handle.net/10217/233704.

Székely, G. J., and M. L. Rizzo, 2005: A new test for multivariate normality. *J. Multivar. Anal.*, **93**, 58–80, https://doi.org/10.1016/j.jmva.2003.12.002.

Van Schaeybroeck, B., and S. Vannitsem, 2015: Ensemble post-processing using member-by-member approaches: Theoretical aspects. *Quart. J. Roy. Meteor. Soc.*, **141**, 807–818, https://doi.org/10.1002/qj.2397.

Vannitsem, S., and Coauthors, 2021: Statistical postprocessing for weather forecasts: Review, challenges, and avenues in a big data world. *Bull. Amer. Meteor. Soc.*, **102**, E681–E699, https://doi.org/10.1175/BAMS-D-19-0308.1.

Veldkamp, S., K. Whan, S. Dirksen, and S. Schmeits, 2021: Statistical postprocessing of wind speed forecasts using convolutional neural networks. *Mon. Wea. Rev.*, **149**, 1141–1152, https://doi.org/10.1175/MWR-D-20-0219.1.

Wan, X., W. Wang, J. Liu, and T. Tong, 2014: Estimating the sample mean and standard deviation from the sample size, median, range and/or interquartile range. *BMC Med. Res. Methodol.*, **14**, 135–, https://doi.org/10.1186/1471-2288-14-135.

Wimmers, A., C. Velden, and J. H. Cossuth, 2019: Using deep learning to estimate tropical cyclone intensity from satellite passive microwave imagery. *Mon. Wea. Rev.*, **147**, 2261–2282, https://doi.org/10.1175/MWR-D-18-0391.1.

Yagli, G. M., D. Yang, and D. Srinivasan, 2022: Ensemble solar forecasting and post-processing using dropout neural network and information from neighboring satellite pixels. *Renewable Sustainable Energy Rev.*, **155**, 111909, https://doi.org/10.1016/j.rser.2021.111909.

Yu, Y., X. Han, M. Yang, and J. Yang, 2020: Probabilistic prediction of regional wind power based on spatiotemporal quantile regression. *IEEE Trans. Ind. Appl.*, **56**, 6117–6127, https://doi.org/10.1109/TIA.2020.2992945.