Identifying Trust Regions of Bayesian Neural Networks

Markus Walker, Marcel Reith-Braun, Peter Schichtel, Mirko Knaak, and Uwe D. Hanebeck

***Abstract*— Bayesian neural networks (BNNs) offer an elegant and promising approach to deciding whether the predictions of a neural network are trustworthy by allowing the estimation of predictive distributions. However, training and prediction can only be performed approximately, and state-of-the-art approximation methods are known to frequently provide inaccurate uncertainty estimations, thus limiting the broad application of neural networks. To remedy this, we define criteria for trustworthy predictions and propose a new approach capable of identifying input space regions with trustworthy predictions. For this, we use statistical hypothesis testing on the BNN's predictions and point out some connections to previously known calibration and uncertainty estimation metrics. We demonstrate our method using several state-of-the-art approximate inference methods on two single-input, single-output regression tasks. Our results show that the proposed approach identifies input space regions with well-calibrated uncertainty predictions while providing valuable insights into the test statistics of the underlying distributions.**

***Index Terms*— Bayesian neural networks, trust regions, uncertainty quantification, calibration, statistical testing.**

I. Introduction

Methods for reasoning and decision making under uncertainty play a pivotal role in developing accurate and interpretable machine learning methods. Artificial neural networks have revolutionized various fields by exhibiting impressive predictive capabilities, but are limited to providing point estimates. In addition, Bayesian Neural Networks (BNNs) introduce a powerful extension to classical neural networks by incorporating a stochastic component, enabling the estimation of the probability distribution over weights and predictions. The Bayesian approach allows to quantify the uncertainty, which is crucial in real-world applications where uncertainties are inherent. The great promise associated with this is the ability to determine how confident a network is in its predictions and to take countermeasures if it is not.

However, while BNNs offer an elegant solution for uncertainty quantification, their practical implementation faces challenges. In general, training and prediction of BNNs lack analytical solutions, necessitating the adoption of various approximation methods, such as Markov Chain Monte Carlo (MCMC) [1], Variational Inference (VI) [2], Expectation Propagation (EP) [3], or Kalman filtering [4]. The accuracy of these approximations is significantly influenced by various model assumptions [5], encompassing the architectural choices and the number of neurons employed. In particular, considerable simplifications are usually assumed for the distribution of the weights, such as a fully factorized normal distribution (known as the mean-field approximation [6]). Furthermore, errors can be caused by suboptimally chosen prior distributions [7] or the quantity of training data [8]. As a consequence, despite contributing to scalability regarding the network's size, these approximation methods are reported to lead to poor and miscalibrated uncertainty predictions, with the latter describing significant mismatches in predicted confidence intervals from observed test points [9].

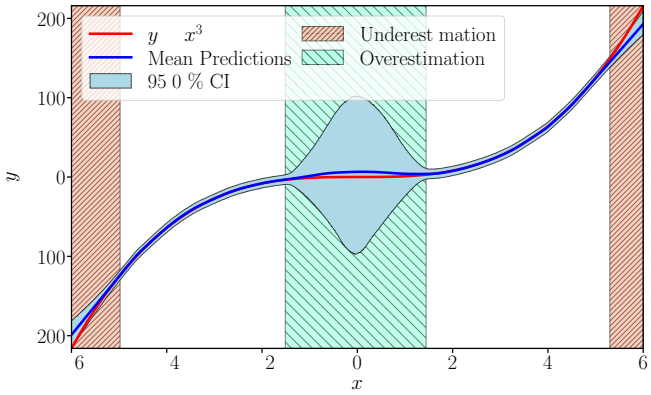


Fig. 1: Identified trust regions of a BNN trained with NUTS using a cubic regression example. The colored areas highlight significant differences ($\alpha = 0.01$) between the BNN's predictions and the test data as evaluated by the ANEES test.

In the BNN literature, there exist various metrics for measuring the quality of the predicted distributions, such as the negative log-likelihood (NLL), the uncertainty calibration error (UCE) [10], the expected normalized calibration error (ENCE) [11], or the quantile calibration error (QCE) [12]. However, these metrics primarily attempt to provide measures for the uncertainty prediction quality using a predefined set of test points, which, e.g., can be used for comparing the prediction quality of different approximate inference methods. A weakness of these measures is that they do not provide help on the question of whether a specific prediction

This work was supported by the German Federal Ministry of Education and Research under grant RobutsNets (FKZ 01IS17042).

Markus Walker, Marcel Reith-Braun, and Uwe D. Hanebeck are with the Intelligent Sensor-Actuator-Systems Laboratory (ISAS), Institute for Anthropomatics and Robotics, Karlsruhe Institute of Technology, Germany (e-mail: markus.walker@kit.edu; marcel.reith-braun@kit.edu; uwe.hanebeck@kit.edu).

Peter Schichtel and Mirko Knaak are with the Ingenieurgesellschaft Auto und Verkehr (IAV) GmbH, Germany (e-mail: peter.schichtel@iav.de; mirko.knaak@iav.de).

may be trustworthy or not. Since this cannot be decided solely on the basis of predicted uncertainties due to their notoriously poor quality, the lack of such methods drastically limits the broad application of neural networks.

In this paper, we explicitly address the question of *whether* one can trust a prediction of a BNN. This is achieved by a slight reformulation of the question under consideration: Instead of asking if the predictions are trustworthy, we ask *when*, or more specifically *in which regions of the input space* the predictions are trustworthy. This change of view acknowledges the fact that there are regions in the input space that are only sparsely covered with training data or not covered at all and may thus, due to lack of training data, never produce appropriate predicted distributions. Our approach thus first identifies regions where the training data is located and then uses rigorous statistical testing to decide whether one can trust the network's predictions within each region (see Fig. 1). Note that our approach is presented using BNNs as an example, but can be applied to Bayesian models in general.

Contributions: First, we provide a definition of what we consider to be trustworthy uncertainty predictions. Based on this, we propose a general strategy to indicate whether the predictions of BNNs are trustworthy consisting of the steps 1.) identifying the input space regions where the data are located, and 2.) checking the calibration within these regions. Second, we present a variant of this strategy that identifies input space regions by evaluating distance measures between an approximate ground truth for the learnable distribution (e.g., using MCMC) and a distribution from a simpler approximate inference approach (e.g., using VI). It then evaluates the calibration within these regions using a parametric test of error distributions or a nonparametric proportion test. Third, we implement this variant for the case of regression tasks in single-input, single-output (SISO) systems, pointing out some connections between statistical hypotheses testing and known metrics for the quality of the uncertainty predictions. We demonstrate the method using two SISO example data sets.

Notation: Throughout this paper, vectors will be indicated by underlined letters, e.g., $\underline{x}$, boldface letters, for instance, $\underline{\boldsymbol{x}}$, will represent random variables, and boldface capital letters, e.g., $\mathbf{A}$, will indicate matrices. The indices in parentheses are used to describe a sorted sequence, e.g., $x_{(1)},x_{(2)},\ldots,x_{(N)}$ represent the ascending sequence of $x_1,x_2,\ldots,x_N$ such that $x_{(1)} \leq x_{(2)} \leq \cdots \leq x_{(N)}$ holds.

II. BNN Learning Setup

We consider a supervised learning setup with a feedforward BNN with L layers. We use the notation $\underline{\boldsymbol{y}} = f(\underline{\boldsymbol{x}},\underline{\boldsymbol{w}})$ for the feedforward BNN, where all weights are summarized in $\underline{\boldsymbol{w}}$ and are considered as random variables with prior distribution $p(\underline{w})$. The data set $\mathcal{D} = \{(\underline{x}_n,\underline{y}_n)\}_{n=1}^{N}$ consists of N independent and identically distributed pairs consisting of the $d_{\underline{x}}$-dimensional input $\underline{x}_n \in \mathbb{R}^{d_{\underline{x}}}$ and $d_{\underline{y}}$-dimensional output $\underline{y}_n \in \mathbb{R}^{d_{\underline{y}}}$. It splits into the training data set $\mathcal{D}_{\text{Train}}$ and the test data set $\mathcal{D}_{\text{Test}}$ with respectively N_{Train} and N_{Test} pairs. The weight posterior distribution $p(\underline{w}\,|\,\mathcal{D}_{\text{Train}})$ as well as the predictive distribution $p(\underline{y}\,|\,\underline{x},\mathcal{D}_{\text{Train}})$ are obtained by

$$p(\underline{w}\,|\,\mathcal{D}_{\text{Train}}) = \frac{p(\mathcal{Y}_{\text{Train}}\,|\,\mathcal{X}_{\text{Train}},\underline{w})p(\underline{w})}{p(\mathcal{Y}_{\text{Train}}\,|\,\mathcal{X}_{\text{Train}})} \ , \tag{1}$$

$$p(\underline{y}\,|\,\underline{x},\mathcal{D}_{\text{Train}}) = \int_{\Omega_{\underline{w}}} p(y\,|\,x,\underline{w})p(\underline{w}\,|\,\mathcal{D}_{\text{Train}})\mathrm{d}\underline{w} \ , \tag{2}$$

where $\mathcal{X}_{\text{Train}} = \{\underline{x}_1, \ldots, \underline{x}_{N_{\text{Train}}}\}$ and $\mathcal{Y}_{\text{Train}} = \{\underline{y}_1, \ldots, \underline{y}_{N_{\text{Train}}}\}$ are the sets of input and output data of the training data set $\mathcal{D}_{\text{Train}}$ and $p(\mathcal{Y}_{\text{Train}}\,|\,\mathcal{X}_{\text{Train}})$ can be considered as a normalization constant. However, there is no general analytical solution to $p(\underline{w}\mid\mathcal{D}_{\text{Train}})$ and $p(\underline{y}\mid\underline{x},\mathcal{D}_{\text{Train}})$. Therefore, in practice, one must resort to approximate inference techniques as outlined in the subsequent section.

III. Related Work

A. Approximate Inference Techniques for BNNs

MCMC [1] provides a popular and powerful approach to probabilistic inference, in particularly for learning in BNNs. This method approximates probability integrals by sampling from a Markov process, which comes with the drawback of high computational cost. Various suggestions for improvement of the standard Metropolis–Hastings algorithm have been proposed, such as Gibbs sampling [13], hybrid Monte Carlo [14], Hamiltonian Monte Carlo (HMC) [15] and its extension, the No-U-Turn Sampler (NUTS) [16]. Note that MCMC approaches represent both $p(\underline{w}\,|\,\mathcal{D}_{\text{Train}})$ and $p(\underline{y}\,|\,\underline{x},\mathcal{D}_{\text{Train}})$ by samples, without making distributional assumptions.

A second similarly popular class of approximate inference algorithms for BNNs is subsumed under the term VI. Here, the general idea is to approximate the complicated weight posterior $p(\underline{w}\mid\mathcal{D}_{\text{Train}})$ by a simpler variational distribution, usually a normal distribution, that allows for a computationally efficient evaluation of (2). The original problem of computing (1) can then be reformulated as an optimization problem w.r.t. the parameters of the variational distribution, which can be solved by minimizing the empirical lower bound of the reverse Kullback–Leibler divergence, usually using gradient descent [2]. In [17] samples from the variational distribution are used in the forward pass to estimate gradients from the entire data set, resulting in gradients with high variance and lack of scalability for larger architectures. To address this issue, Stochastic VI (SVI) [6] utilizes scaled gradients from randomly chosen subsets of training data to update the variational distribution, thus improving scalability. A deterministic approach is proposed in [18] that utilizes approximate moment propagation instead of sampling in the forward pass, resulting in reduced gradient variance. To reduce computational costs, [19], [20] proposed the use of the dropout technique, which approximates the variational distribution [21]. Note that $p(\underline{w}\mid\mathcal{D}_{\text{Train}})$

is described by the variational distribution (usually a normal distribution), where neuron-wise, layer-wise, or network-wide correlations can be realized [2].

Building upon the same ideas as VI, EP [3] minimizes the forward Kullback–Leibler divergence rather than the reverse Kullback–Leibler divergence. EP's versions for BNNs, such as Probabilistic Backpropagation (PBP) [22] and its extension [23], both utilize the mean-field approximation, closed-form moment propagation during the forward pass and gradient-based weight updates in the backward pass.

In addition to Monte Carlo approximations or methods that require explicit computation of gradients for optimization, Kalman filtering approaches can be used to train BNNs. These approaches assume normally distributed weights and differ in the number of correlations considered and the way nonlinearities are accounted for. For example, the extended Kalman filter [4], [24], which requires analytical linearizations of the network's nonlinearities, or sampling-based Kalman filtering techniques such as unscented Kalman filter (UKF) [25] or the ensemble Kalman filter [26] were used. To avoid linearization or sampling, [27] proposes the Bayesian perceptron, which analytically calculates the mean and covariance of $p(\underline{w}\,|\,\mathcal{D}_{\text{Train}})$ and $p(\underline{y}\,|\,\underline{x},\mathcal{D}_{\text{Train}})$ with correlated weights for certain kinds of activation functions. [28] extended the Bayesian perceptron to a fully connected feedforward network with neuron-wise correlated weights, referred to as Kalman BNN (KBNN).

B. Statistical Hypothesis Testing

Statistical hypothesis testing is a fundamental part of inferential statistics, which is used to make decisions about hypotheses based on data using a test statistic T. The process involves testing a null hypothesis, denoted H_0, against its alternative hypothesis H_1. If the data provides significant evidence against H_0, then H_0 is rejected in favor of H_1. During a test procedure, two kinds of errors can occur. The error of the first kind occurs when a true H_0 is incorrectly rejected, whereas the error of the second kind occurs when the test fails to reject an incorrect H_0. H_0 is rejected if the test statistic T is less than a lower critical value c_l or greater than an upper critical value c_u, for two-sided tests. These critical values are designed to limit the probability of error of the first kind α. Typically, α is set to $0.01 \leq \alpha \leq 0.1$, ensuring that a false rejection of H_0 will occur in no more than $100 \cdot \alpha$ % of cases. Statistical tests can be divided into parametric and nonparametric. Parametric tests are associated with certain assumptions about the distribution of the test data, e.g., normality, while nonparametric tests do not impose strict assumptions, thus providing more flexibility. For more detailed descriptions of the principles of statistical testing, we refer to [29]. In the following, we briefly review the tests that will be used in the presentation of our approach.

1) Parametric Tests: If the data is normally distributed, well-known parametric tests such as Student's t-test or F-tests can be used to test for either matching means or variances [29]. To determine whether the data are consistent with an estimated normal distribution, the chi-square test w.r.t. the average squared Mahalanobis distance, also known as the averaged normalized estimation error squared (ANEES) [30], as a test statistic can be used. This test will be referred to as the ANEES test, with its test statistic T_{ANEES} given by

$$T_{\text{ANEES}} = \frac{1}{S}\sum_{s=1}^{S}\left(\underline{y}_s - \underline{\mu}_s^{\underline{\boldsymbol{y}}}\right)^T \left(\mathbf{C}_s^{\underline{\boldsymbol{y}}}\right)^{-1}\left(\underline{y}_s - \underline{\mu}_s^{\underline{\boldsymbol{y}}}\right) \quad , \tag{3}$$

where S is the number of considered normalized estimation errors squared (NEES), $\underline{\mu}_s$ is the estimated mean, $\mathbf{C}_s^{\underline{\boldsymbol{y}}}$ is the estimated covariance matrix and $\underline{y}_s$ is a data sample of dimension d [30]. If the normality assumption holds, the ANEES follows a chi-squared distributed test statistic, resulting in the critical values of the test statistic given by

$$[c_\text{l}, c_\text{u}] = \frac{1}{S}\left[F^{-1}_{\chi^2_k}\left(\frac{\alpha}{2}\right),\ F^{-1}_{\chi^2_k}\left(1-\frac{\alpha}{2}\right)\right] \quad , \tag{4}$$

with $F^{-1}_{\chi^2_k}$ being the chi-square inverse cumulative distribution function with $k = d \cdot S$ degrees of freedom. Falling below the lower critical value c_l indicates that the uncertainty of the estimated normal distribution is larger than the data reflect. Exceeding the upper critical value c_u indicates a significant bias or an uncertainty that is too small [30]. Note that the higher S is, the smaller the resulting interval $[c_\text{l}, c_\text{u}]$, reflecting the reduction in variability with repeated simulations [30].

2) Nonparametric Tests: If the test data is not assumed to follow a specific distribution, nonparametric tests such as the famous Kolmogorov–Smirnov (KS) test [31], calculating the maximum distance between two cumulative distributions, can be used. Furthermore, the Cramér–von Mises test [32] or the Anderson–Darling test [33], which use the squared distances between two cumulative distributions or their weighted distances, respectively, are applicable. If, instead of cumulative distributions, a ratio π between two possible categories of outcomes of a Bernoulli experiment is to be tested, the binomial test can be used. Therefore, the proportion π is the ratio k/S, where k describes the successes belonging to one possible category of the experiment in S trials. Under $H_0 \colon \pi = \pi_0$, this results in a binomially distributed test statistic $T_{\text{Bin}} \sim \text{Bin}(S, \pi_0)$. The test statistic value T_{Bin} is obtained by

$$T_{\text{Bin}} = \text{Bin}(k\,|\,S,\pi_0) = \binom{S}{k} \pi_0^k (1-\pi_0)^{S-k} \quad , \tag{5}$$

with k successes in S trials. Instead of critical values of the test statistics, the critical values of successes, k_l and k_u can be determined such that

$$\sum_{k=0}^{k_\text{l}} \text{Bin}(k\,|\,\pi_0,S) \leq \frac{\alpha}{2} \quad , \quad \sum_{k=k_\text{u}}^{S} \text{Bin}(k\,|\,\pi_0,S) \leq \frac{\alpha}{2} \tag{6}$$

applies in order to perform two-sided tests. Thus, for example, if the number of hits k is less than k_l or greater

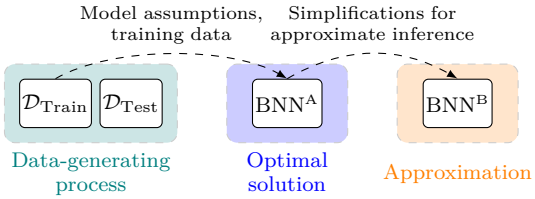


Fig. 2: Illustration of our model of involved distributions in learning.

than k_{u}, then H_0 is rejected because there is significant evidence against the proportion π_0. When applying the binomial test, it is crucial to respect the minimum sample size required to detect certain population deviations $\pm\epsilon$ from the population to be tested π_0. An estimation of the minimum sample size required can be obtained based on the Hoeffding's inequality [34] or the more conservative Chebyshev inequality resulting in $s_{\text{min}}^{\text{Hoeff.}} \geq \frac{\ln(\alpha/2)}{2\epsilon^2}$ and $s_{\text{min}}^{\text{Cheb.}} \geq \frac{\pi_0(1-\pi_0)}{\epsilon^2\alpha}$, respectively.

C. Uncertainty Calibration Measures for Regression Tasks

The quality of uncertainty estimates in BNN regression tasks is usually measured using scoring rules that either evaluate the combined prediction and uncertainty estimation quality, such as the NLL, or the calibration of uncertainty estimates. Calibration in this context refers to how well the uncertainty estimates match the observed errors between prediction and actual values on test data points. However, there are several ways of measuring calibration. For one-dimensional normally distributed predictions, recently proposed calibration measures such as UCE [10] and ENCE [11] compare predicted variances with the observed mean squared error, exploiting the relationship between both. For one- and multivariate normally distributed predictions [12] proposed the QCE, which compares the observed frequencies and the desired quantile values of the chi-squared distributed errors. It is worth noting that the QCE uses the same error distribution as the ANEES test, but not for statistical testing. For non-normal univariate predictions, the pinball loss [35], which compares certain quantiles can be employed. Additionally, calibration plots [36] can be used for a visual comparison of how accurately expected and observed confidence levels match across all test data.

IV. Trust Region Identification

In this section, we examine the distributions involved in learning, and use this model to create a general testing strategy.

A. A General Model of Distributions in Learning

Our model is built on three distributions to explain the learning environment and is inspired by but not identical to the one in [8] (see Fig. 2). The first distribution $p(\underline{y}\,|\,\underline{x})$ reflects the true data-generating process that produces the observed outcomes given inputs $\underline{x}$. Unfortunately, this distribution is unknown and only realizations of input-output pairs are available in the data set $\mathcal{D}$. The second distribution $p_n^{\text{A}}(\underline{y}\,|\,\underline{x}_n,\mathcal{D}_{\text{Train}})$ constitutes the (optimal, but infeasible) solution to (2), given a fixed family of regression functions $f(\underline{\boldsymbol{x}},\underline{\boldsymbol{w}})$, i.e., the BNN's architecture, the prior distribution for the weights $p(\underline{w})$, and a training data set $\mathcal{D}_{\text{Train}}$. A BNN that corresponds to $p_n^{\text{A}}(\underline{y}\,|\,\underline{x}_n,\mathcal{D}_{\text{Train}})$ is denoted as BNN$^{\text{A}}$. Note, however, that it is not guaranteed that BNN$^{\text{A}}$ will map the data-generating process well [8], since its modeling abilities depend on the choice of the BNN's architecture, the prior weight distribution, and the training data. The third distribution $p_n^{\text{B}}(\underline{y}\,|\,\underline{x}_n,\mathcal{D}_{\text{Train}})$, represented by a BNN instance BNN$^{\text{B}}$, is the result of an approximative inference technique on (2), performed by methods such as those presented in Sec. III-A. In addition to the errors made when deciding on model architecture, prior weight distribution, and training data set collection, this distribution also incorporates errors from the approximation of (1) and (2), i.e., the training and inference procedures.

B. A Definition for Trustworthy Uncertainty Predictions

Based on these three distributions, ideally, we would like to test a BNN against the true distribution $p(\underline{y}\,|\,\underline{x})$ to decide whether the BNN's predicted distributions are reliable. However, since $p(\underline{y}\,|\,\underline{x})$ is unknown, we need a practical yet wider definition of what is meant by trustworthy uncertainty predictions. Our definition provided below builds upon the following two criteria:

1.) **Proximity to training data:** A major challenge is that complete continuous support in the input space cannot be covered by the finite sampled data. Consequently, there are regions in the input space that are densely covered with data, while there are other regions that are only sparsely covered or not covered at all. Thus, also an appropriate choice of a network's architecture and prior weights distribution will not guarantee that an optimally trained BNN $p^{\text{A}}(\underline{y}\,|\,\underline{x},\mathcal{D}_{\text{Train}})$ matches $p(\underline{y}\,|\,\underline{x})$ for all $\underline{x}$. In fact, it is totally hidden to us what may happen in regions not covered by the available data (note that also the prior distribution $p(\underline{w})$ does not provide help to this question). It is our understanding that, for trustworthy predictions, one should thus avoid extrapolation and stay "close" to the provided training data. By "close" in this context, we mean the input space regions where an appropriately designed and optimally trained BNN$^{\text{A}}$ should be able to learn the distribution from the data, i.e., the input space regions where proximity is ensured are those where $p^{\text{A}}(\underline{y}\,|\,\underline{x},\mathcal{D}_{\text{Train}})$ and $p(\underline{y}\,|\,\underline{x})$ match.

2.) **Calibrated predictions:** In the case that proximity is ensured, errors may still occur in BNN$^{\text{A}}$ due to an inappropriate choice of architecture or prior. Likewise, approximate inference may lead to additional errors in BNN$^{\text{B}}$. Consequently, a meaningful prediction must be "calibrated". By "calibrated'' in this context, we mean that the predicted distribution must accurately reflect $p(\underline{y}\,|\,\underline{x})$ through the sampled test data, i.e., the predicted distribution of BNN$^{\text{B}}$ and the empirical distribution of the sampled test data in these regions must match.

C. A General Test Strategy Using Candidate Regions

Note that by the definition of proximity, if a BNN is calibrated within a certain input region, proximity is also satisfied within this region. Therefore, it is immaterial whether it constitutes the optimal solution BNN^{A} or an approximation BNN^{B}, since both models are susceptible to errors. A testing strategy based on this consideration may therefore exploit that is sufficient to test the calibration of the predictions of BNN^{A} or BNN^{B} in a certain region of interest using samples of the test data set. A general strategy may therefore consist of the steps: 1) Identify input space region candidates where data is located, and 2) verify calibration within these regions. Note, however, that while this strategy is simple, it does not guarantee that all regions with sufficient input data will be identified.

D. Our Proposed Variant

We now propose a variant of the general testing strategy presented in the previous subsection (see Fig. 3 for an illustration). The basic concept to identify candidate input space regions in this variant is to compare two trained BNN distributions $p_n^{\tilde{\text{A}}}(\underline{y}\,|\,\underline{x}_n,\mathcal{D}_{\text{Train}})$ and $p_n^{\text{B}}(\underline{y}\,|\,\underline{x}_n,\mathcal{D}_{\text{Train}})$ at the inputs $\underline{x}_n \in \mathcal{X}_{\text{Test}}$ of the test data set $\mathcal{D}_{\text{Test}} = (\mathcal{X}_{\text{Test}},\mathcal{Y}_{\text{Test}})$ using a distance measure for probability distributions. Here, $p_n^{\tilde{\text{A}}}(\underline{y}\,|\,\underline{x}_n,\mathcal{D}_{\text{Train}})$ denotes an approximation to $p_n^{\text{A}}(\underline{y}\mid\underline{x}_n,\mathcal{D}_{\text{Train}})$ since $p_n^{\text{A}}(\underline{y}\mid\underline{x}_n,\mathcal{D}_{\text{Train}})$ is infeasible. For $p_n^{\tilde{\text{A}}}(\underline{y}\,|\,\underline{x}_n,\mathcal{D}_{\text{Train}})$, one may e.g. use MCMC or any other method that is known to provide an accurate approximation. The idea is based on the assumption that in extrapolation the predicted distributions of $\text{BNN}^{\tilde{\text{A}}}$ and BNN^{B} differ significantly (which can be detected by distance measures for probability distributions) due to the lack of training data, thus directly pointing to regions where no training data were present.

Given the input data and their associated distance values, the input data that lead to similar distance values are then grouped, creating candidate input space regions (step 1). This step can be seen as a form of semi-supervised clustering [37]. Once the input areas have been identified, we use statistical tests to assess how accurately the predicted outputs of each region match the outputs of the test data set (step 2).

V. Special Case: SISO Systems

In this section, we apply our proposed testing strategy (cf. Sec. IV-D) to the special case of SISO systems.

A. Candidate Region Identification

We assume that each prediction $p_n^{\tilde{\text{A}}}(\underline{y}\mid\underline{x}_n,\mathcal{D}_{\text{Train}})$ of $\text{BNN}^{\tilde{\text{A}}}$ (given by MCMC) for a test input $x_n \in \mathcal{X}_{\text{Test}}$ is represented by a Dirac mixture $\frac{1}{R}\sum_{r=1}^{R}\delta(y-y_r)$. The predictions of BNN^{B}, $p_n^{\text{B}}(\underline{y}\,|\,\underline{x}_n,\mathcal{D}_{\text{Train}})$, can be either expressed as a Dirac mixture $\frac{1}{M}\sum_{m=1}^{M}\delta(y-y_m)$ (e.g., if BNN^{B} employs SVI or a MCMC algorithm), or as a normal distribution $\mathcal{N}(\mu_n^{\boldsymbol{y}},(\sigma_n^{\boldsymbol{y}})^2)$ with mean $\mu_n^{\boldsymbol{y}}$ and variance $(\sigma_n^{\boldsymbol{y}})^2$ (e.g., if PBP, or KBNN is used). For identifying possible candidate regions, we thus calculate the distances between

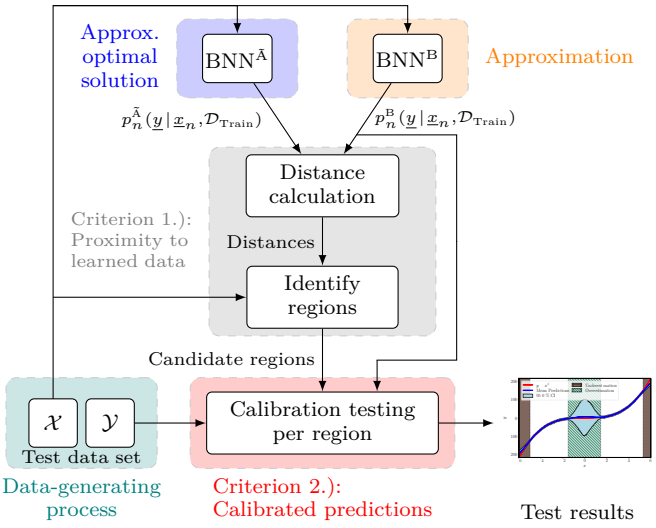


Fig. 3: Illustration of our proposed testing strategy. The strategy compares two BNN models, $\text{BNN}^{\tilde{\text{A}}}$ and BNN^{B}, to find candidate regions in the input space using distance measures for probability distributions. It then uses statistical calibration testing to identify trustworthy regions within the identified candidate regions.

$p_n^{\tilde{\text{A}}}(\underline{y}\,|\,\underline{x}_n,\mathcal{D}_{\text{Train}})$ and $p_n^{\text{B}}(\underline{y}\,|\,\underline{x}_n,\mathcal{D}_{\text{Train}})$ employing the 1-Wasserstein distance. If $p_n^{\text{B}}(y\,|\,x_n,\mathcal{D}_{\text{Train}})$ is represented by a Dirac mixture, it can be calculated according to [38]

$$d_n(p_n^{\tilde{\text{A}}},p_n^{\text{B}}) = \min_{t_{rm}} \sum_{r=1}^{R}\sum_{m=1}^{M} t_{rm}\|y_r - y_m\|$$
$$\text{s.t.} \sum_{r}^{R} t_{rm} = \frac{1}{M} \quad \forall j \ , \quad \sum_{m}^{M} t_{rm} = \frac{1}{R} \quad \forall m \ ,$$

where $t_{nm} \geq 0$ describes the amount of probability mass that is moved between the two samples r and m. When $p_n^{\text{B}}(\underline{y}\,|\,\underline{x}_n,\mathcal{D}_{\text{Train}})$ is represented by a normal distribution, the 1-Wasserstein distance is obtained by [39]

$$d_n(p_n^{\tilde{\text{A}}},p_n^{\text{B}}) = \frac{1}{R}\sum_{r=1}^{R}\left| F_{\mathcal{N}}^{-1}\left(\frac{\tau_r+\tau_{r-1}}{2}\right) - y_{(r)}\right| \ ,$$

where $F_{\mathcal{N}}^{-1}$ is the inverse cumulative distribution function of $p_n^{\text{B}}(\underline{y}\mid\underline{x}_n,\mathcal{D}_{\text{Train}})$, and $y_{(r)}$ is the r-th lowest value of $p_n^{\tilde{\text{A}}}(\underline{y}\mid\underline{x}_n,\mathcal{D}_{\text{Train}})$ with corresponding quantile level $\tau_r = \frac{r}{R}$. In both cases, we obtain the set $\mathcal{I} = \{(x_n,d_n)\}_{n=1}^{N}$ consisting of the test inputs values x_n and associated distance values d_n.

To identify regions, we use a grouping algorithm that first orders the set $\mathcal{I}$ according to x_n to account for the proximity of the input values and additionally calculates the sorted sequence of the distance values $d_{(n)}$. A critical distance value d_{crit} is then used to divide the input space at index $j = n$ if $d_n > d_{\text{crit}}$. To reduce errors of the first and second kind of statistical tests in this step, d_{crit} is based on the minimum number of test data points per region, $s_{\min}$ (cf. Sec. III-B). Since the adjacent distance values can fluctuate, we use a moving average filter with a window size of

s_{avg} to smooth them in order to avoid frequent region divisions close to the critical value. The resulting regions are then described by $\mathcal{X}_k = \{x \,|\, x_j \leq x < x_{j+1}\}$, where x_0 is the lower bound of the first region, and x_N is the upper bound of the last region. Since the algorithm described above is very sensitive to its parameter d_{crit}, we use a heuristic for finding a suitable critical value. For this, we initialize the critical value with $d_{\text{crit}} = d_{(1)}$ and subsequently increase it to the next highest distance value until all identified regions contain at least $s_{\min}$ test points. If the number of test points in an interval s_k is at least twice as large as $s_{\min}$, we additionally divide the region $\mathcal{X}_k$ into $\left\lfloor \frac{s_k}{s_{\min}} \right\rfloor$ parts of similar size to create a finer subdivision of the input space. Note that the first and last region may contain fewer data points than $s_{\min}$. This ensures that outliers in $x_n \in \mathcal{X}_{\text{test}}$ do not lead to an unwanted increase of d_{crit} and individual outliers are not assigned to neighboring candidate regions that are densely covered with test data.

B. Statistical Calibration Testing

Here, we present two possibilities for testing the calibration of $p_n^a(y \,|\, x_n, \mathcal{D}_{\text{Train}})$ with $a \in \{\tilde{\text{A}}, \text{B}\}$ in each interval $\mathcal{X}_k$ using either a parametric or a nonparametric statistical test. The parametric test is suitable when both the prediction and output are approximately normally distributed, while the nonparametric test is appropriate when the normality assumption does not hold, or one only is interested in whether the confidence intervals of a certain confidence level are sufficiently estimated.

As a parametric test, we use the ANEES test and calculate the ANEES value and the critical values of the test statistic for each identified input region using the equations (3) and (4). If the ANEES value is lower than the critical lower bound, this suggests that the BNN has overestimated the uncertainty in the corresponding input region. Conversely, if the calculated ANEES value is greater than the critical upper bound, this indicates that the BNN has underestimated the uncertainty or has a significant bias in the corresponding input region. If the ANEES value is within the bounds, we do not reject H_0 since we do not have significant evidence against H_0. To additionally check the validity of the initial assumption of normality, we use the nonparametric KS test, i.e., we check whether the NEES values follow the assumed chi-square distribution.

As a nonparametric test, we use the two-sided binomial test (5) and (6) to check whether the frequency of $y_n \in \mathcal{Y}_k$ within the predicted confidence interval of $p_n^a(y \,|\, x_n, \mathcal{D}_{\text{Train}})$ matches the confidence interval π_0 being tested. E.g., if $\pi_0 = 0.95$ of a region is to be examined, the binomial test is used to check whether 95 % of $y_n \in \mathcal{Y}_k$ lie within the 95 % confidence interval of the predictions of the region k. Again, if the bounds (6) are respected, we do not reject H_0.

VI. Numerical Evaluation

We now demonstrate the proposed SISO testing strategy on two synthetic data sets.

A. BNN Architecture, Training, and Test Parameters

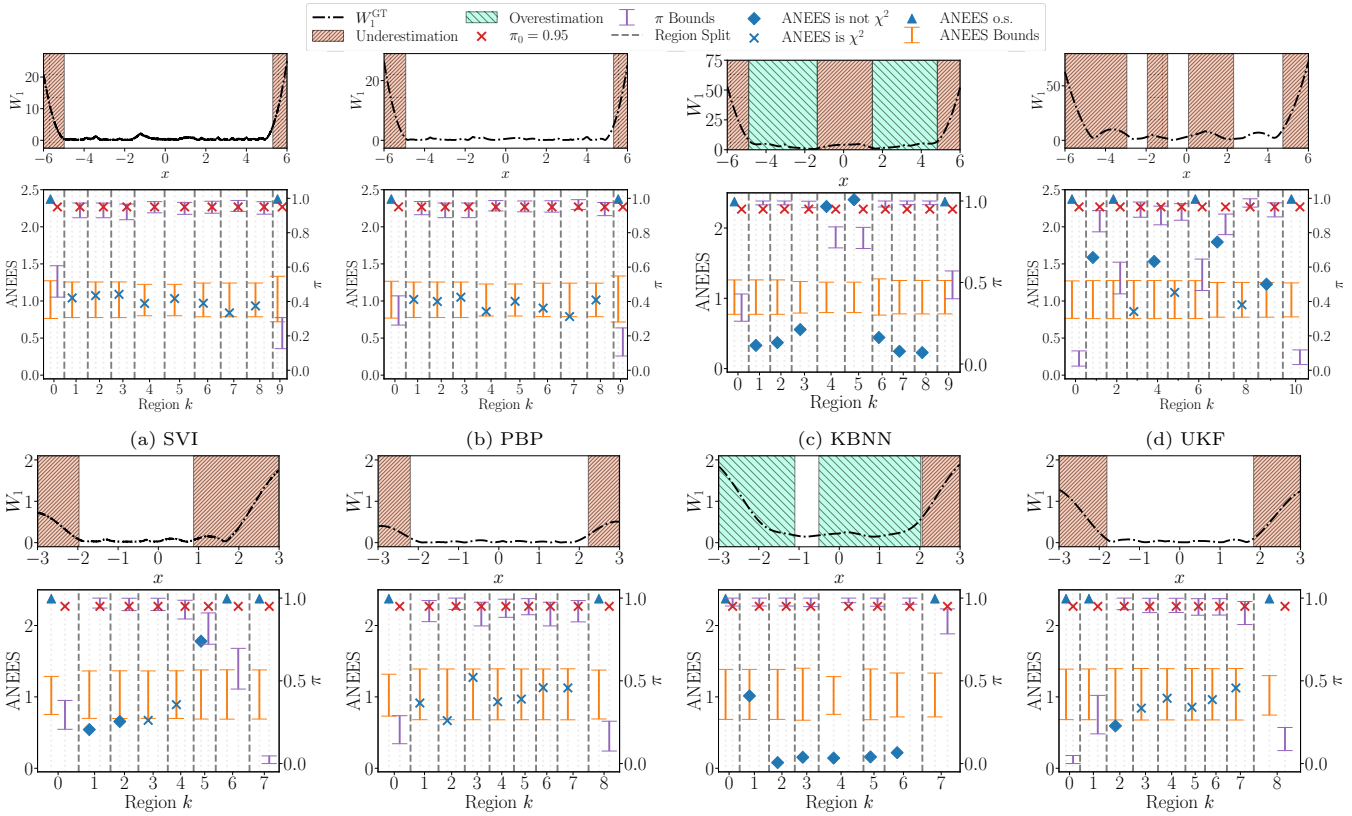
For all experiments, we use a fully connected feedforward network with one hidden layer, 50 hidden neurons, ReLU activation for the hidden layer, and linear activation for the output layer. We use NUTS [16] for $\text{BNN}^{\tilde{\text{A}}}$, and for BNN^{B} we test variants with SVI [6] and PBP [22] with mean-field approximation, KBNN [28] with neuron-wise correlated weights, and UKF [25] with network-wide correlated weights. For NUTS and SVI, we use implementations provided by the probabilistic programming package NumPyro [40]. For PBP [22] and KBNN [28], we use the implementations of the authors. The implementation of UKF for BNNs is realized following the procedure of [25].

We train the BNNs using SVI, PBP, KBNN, and UKF for 80000, 200, 10, and 1 epochs, respectively. Statistical tests are conducted with a significance level of $\alpha = 0.01$. We use $\pi_0 = 0.95$ for the two-sided binomial test, which corresponds to a tested prediction confidence interval of 95 %. To identify regions, a moving average filter with a window size of $s_{\text{avg}} = 50$ is used to obtain smooth adjacent distance values.

B. Experiments

For our first experiment, we generate 2000 training points and 2400 test points from $\boldsymbol{y} = x^3 + \boldsymbol{\epsilon}$, with $\boldsymbol{\epsilon} \sim \mathcal{N}(0, 9)$. Training inputs $x_n \in \mathcal{X}_{\text{Train}}$ and test inputs $x_n \in \mathcal{X}_{\text{Test}}$ are drawn uniformly from $[-5,5]$ and $[-6,6]$, respectively. We then remove the center 30 % of the training data points to simulate a gap in the data. To identify input space regions, $s_{\min}$ is set to 200, which is greater than the minimum sample size $s_{\min}^{\text{Hoeff.}}$ required for the binomial test according to Hoeffding's inequality (see Sec. III-B.2). The results of all approximate inference methods tested are shown in Fig. 4a-d. The predictions with NUTS as $\text{BNN}^{\tilde{\text{A}}}$ are shown in Fig. 1. The input space regions $x < -5$ and $x > 5$ in which no training data were available show biased and miscalibrated predictions and were consequently detected by both statistical tests. Interestingly, for SVI and PBP, the center regions lead to calibrated predictions, although no training data were available. In contrast, for KBNN and UKF, e.g., in the region $k = 4$, the ANEES test detects overconfident or biased predictions. Similarly, for the KBNN, both statistical tests detect overestimated uncertainties in the regions $k = 1$ and $k = 2$.

As a second experiment, we train the BNNs on 500 uniformly distributed samples drawn from $\boldsymbol{y} = x + \sin^2(x) + \boldsymbol{\epsilon}$, $\boldsymbol{\epsilon} \sim \mathcal{N}(0, 0.1)$ in $x \in [-2, 2]$ and test the predictions using 1000 test inputs ranging from -3 to 3. To identify the input space regions $s_{\min}$ is set to 100 since we have less data than in the first example. The results are shown in Fig. 4e-h. As in our first example, all regions where training data were not available are detected. Again, test statistics and critical values differ significantly



(a) SVI (b) PBP (c) KBNN (d) UKF

(e) SVI (f) PBP (g) KBNN (h) UKF

Fig. 4: Results of the candidate region identification and the statistical calibration testing. The results of the first experiment $\boldsymbol{y}=x^3+\boldsymbol{\epsilon}$ are depicted in a-d. The results of the second experiment $\boldsymbol{y}=x+\sin^2(x)+\boldsymbol{\epsilon}$ are shown in e-h. The 1-Wasserstein distance between the predictions and the true data-generating process (denoted by W_1^{GT}) is shown in each upper plot. In the same plot, calibration test decisions are highlighted by the colored areas (using the ANEES test for a-d and the two-sided binomial test for e-h). The lower plot displays the identified candidate regions as well as test statistics for each input region of both the ANEES and the two-sided binomial test. In case the blue ANEES markers do not overlap with the orange confidence bar illustrating the range between the critical values, H_0 is rejected. The same applies to the red markers and their purple displayed confidence ranges in the case of the two-sided binomial test. If the KS test rejects H_0, the ANEES markers are represented by diamonds, otherwise, by crosses. Triangular-shaped ANEES markers represent values that are outside the scope (ANEES o.s.) of the plot and lead to the rejection of H_0.

in these regions. It is noticeable that the ANEES test in the region $k=1$ of the KBNN does not reject H_0, although W_1^{GT} is similar to the $k=1$ region of the UKF, where the ANEES test rejects the region. However, as the result of the KS test shows, the test assumption of normality is violated in this case, indicating that the ANEES test is not applicable in this region of the KBNN.

C. Discussion

The results show that the 1-Wasserstein distance is highly sensitive to differences in the compared distributions and thus an appropriate choice for candidate region identification. The region identification itself is able to divide the input space into regions of sufficient number of data points. Both statistical tests are able to detect cases where the predictions were strongly biased, either under- or overconfident. The ANEES test uses the error distribution between predictions and data, which leads to a stronger statement regarding calibration. However, its dependence on the normality assumption is a major limitation. In contrast, the binomial test does not assume normality, but only one confidence interval is evaluated. For almost all intervals the test results of the ANEES and two-sided binomial are identical. However, small deviations of the predictions from the ground truth, expressed by their Wasserstein distances, such as those in the input space region $k=3$ in Fig. 4 (g) can lead to different test decisions for the ANEES test and the binomial test. In such cases, special attention has to be paid to the test results, since they, by definition, incorrectly reject H_0 in up to $100\cdot\alpha$ %. Here, statistics and critical values as well as their distances can provide useful information beyond binary decisions per input space region.

VII. Conclusion

This paper introduced a testing strategy to locate trustworthy input space regions in BNNs, i.e., input space regions that lead to predicted distributions accurately reflected in test data. For this, we first detected candidate regions and then used the parametric ANEES test and non-parametric binomial tests to assess the prediction quality

in these regions. Our approach was able to accurately detect input space regions with trustworthy uncertainty prediction for our two considered SISO examples. Moreover, the approach revealed under- or overconfident input space regions and established a foundation for refining Bayesian model evaluation. Therefore, we believe that it can be considered as a first step towards an approval process for Bayesian models such as BNNs and may help to open a wider field of application for neural networks in general.

Future research directions include developing a method to identify candidate regions without requiring a second trained Bayesian model and extending the region identification to multiple-input, multiple-output systems.

References

[1] N. Metropolis *et al.*, "Equation of State Calculations by Fast Computing Machines," *J. Chem. Phys.*, vol. 21, no. 6, pp. 1087–1092, 1953.

[2] A. Graves, "Practical variational inference for neural networks," in *NIPS'11: Proceedings of the 24th International Conference on Neural Information Processing Systems*, 2011, pp. 2348–2356.

[3] T. P. Minka, "A family of algorithms for approximate Bayesian inference," Ph.D. dissertation, Massachusetts Institute of Technology, 2001.

[4] K. Watanabe and S. G. Tzafestas, "Learning algorithms for neural networks with the Kalman filters," *Journal of Intelligent and Robotic Systems*, vol. 3, no. 4, pp. 305–319, 1990.

[5] P. Izmailov *et al.*, "What Are Bayesian Neural Network Posteriors Really Like?" in *Proceedings of the 38th International Conference on Machine Learning*, 2021, pp. 4629–4640.

[6] M. D. Hoffman *et al.*, "Stochastic variational inference," *J. Mach. Learn. Res.*, vol. 14, no. 1, pp. 1303–1347, 2013.

[7] V. Fortuin *et al.*, "Bayesian Neural Network Priors Revisited," in *International Conference on Learning Representations*, 2022.

[8] L. Bottou and O. Bousquet, "The tradeoffs of large scale learning," in *Proceedings of the 20th International Conference on Neural Information Processing Systems*, 2007, pp. 161–168.

[9] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 6405–6416.

[10] M.-H. Laves *et al.*, "Well-Calibrated Regression Uncertainty in Medical Imaging with Deep Learning," in *Proceedings of the Third Conference on Medical Imaging with Deep Learning*, vol. 121, 2020, pp. 393–412.

[11] D. Levi *et al.*, "Evaluating and Calibrating Uncertainty Prediction in Regression Tasks," *Sensors*, vol. 22, no. 15, 2022.

[12] F. Küppers, J. Schneider, and A. Haselhoff, "Parametric and Multivariate Uncertainty Calibration for Regression and Object Detection," in *Computer Vision – ECCV 2022 Workshops*, 2023, vol. 13805, pp. 426–442.

[13] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 6, no. 6, pp. 721–741, 1984.

[14] R. M. Neal, "Bayesian Learning for Neural Networks," Ph.D. dissertation, University of Toronto, 1995.

[15] S. Duane *et al.*, "Hybrid Monte Carlo," *Physics Letters B*, vol. 195, no. 2, pp. 216–222, 1987.

[16] M. D. Homan and A. Gelman, "The No-U-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1593–1623, 2014.

[17] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," in *The 2nd International Conference on Learning Representations*, 2013.

[18] A. Wu *et al.*, "Deterministic Variational Inference for Robust Bayesian Neural Networks," in *International Conference on Learning Representations*, 2019.

[19] N. Srivastava *et al.*, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014.

[20] Y. Gal, J. Hron, and A. Kendall, "Concrete dropout," in *Advances in neural information processing systems*, vol. 30, 2017.

[21] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning," in *Proceedings of The 33rd International Conference on Machine Learning*, vol. 48, 2016, pp. 1050–1059.

[22] J. M. Hernández-Lobato and R. P. Adams, "Probabilistic backpropagation for scalable learning of Bayesian neural networks," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, 2015, pp. 1861–1869.

[23] S. Ghosh, F. M. D. Fave, and J. Yedidia, "Assumed density filtering methods for learning Bayesian neural networks," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 2016, pp. 1589–1595.

[24] G. V. Puskorius and L. A. Feldkamp, "Parameter-Based Kalman Filter Training: Theory and Implementation," in *Kalman Filtering and Neural Networks*, 2001, pp. 23–67.

[25] E. Wan and R. Van Der Merwe, "The unscented Kalman filter for nonlinear estimation," in *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium*, 2000, pp. 153–158.

[26] C. Chen *et al.*, "Approximate Bayesian Neural Network Trained with Ensemble Kalman Filter," in *2019 International Joint Conference on Neural Networks (IJCNN)*, 2019, pp. 1–8.

[27] M. F. Huber, "Bayesian Perceptron: Towards fully Bayesian Neural Networks," in *59th IEEE Conference on Decision and Control (CDC)*, 2020, pp. 3179–3186.

[28] P. Wagner, X. Wu, and M. F. Huber, "Kalman Bayesian Neural Networks for Closed-form Online Learning," in *37th AAAI Conference on Artificial Intelligence*, 2023.

[29] E. Lehmann and J. P. Romano, *Testing Statistical Hypotheses*. Springer International Publishing, 2022.

[30] Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. John Wiley & Sons, Inc., 2001.

[31] N. Smirnov, "Table for Estimating the Goodness of Fit of Empirical Distributions," *The Annals of Mathematical Statistics*, vol. 19, no. 2, pp. 279–281, 1948.

[32] T. W. Anderson, "On the Distribution of the Two-Sample Cramer-von Mises Criterion," *The Annals of Mathematical Statistics*, vol. 33, no. 3, pp. 1148–1159, 1962.

[33] T. W. Anderson and D. A. Darling, "Asymptotic Theory of Certain "Goodness of Fit" Criteria Based on Stochastic Processes," *The Annals of Mathematical Statistics*, vol. 23, no. 2, pp. 193–212, 1952.

[34] W. Hoeffding, "Probability Inequalities for Sums of Bounded Random Variables," *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 13–30, 1963.

[35] I. Steinwart and A. Christmann, "Estimating conditional quantiles with the help of the pinball loss," *Bernoulli*, vol. 17, no. 1, 2011.

[36] V. Kuleshov, N. Fenner, and S. Ermon, "Accurate uncertainties for deep learning using calibrated regression," in *Proceedings of the 35th international conference on machine learning*, vol. 80, 2018, pp. 2796–2804.

[37] Y. Qin *et al.*, "Research Progress on Semi-Supervised Clustering," *Cognitive Computation*, vol. 11, no. 5, pp. 599–612, 2019.

[38] D. Schuhmacher, B.-T. Vo, and B.-N. Vo, "A Consistent Metric for Performance Evaluation of Multi-Object Filters," *IEEE Transactions on Signal Processing*, vol. 56, no. 8, pp. 3447–3457, 2008.

[39] Y. Chen, X. Hou, and Y. Liu, "Minimizing Wasserstein-1 Distance by Quantile Regression for GANs Model," in *Pattern Recognition and Computer Vision*, 2021, pp. 128–139.

[40] D. Phan, N. Pradhan, and M. Jankowiak, "Composable effects for flexible and accelerated probabilistic programming in NumPyro," *arXiv preprint arXiv:1912.11554*, 2019.