EPISTEMIC WRAPPING FOR UNCERTAINTY QUANTIFICATION

Maryam Sultana
Oxford Brookes University, UK
msultana@brookes.ac.uk

Neil Yorke-Smith
Delft University of Technology, The Netherlands
n.yorke-smith@tudelft.nl

Kaizheng Wang
KU Leuven University, Belgium
kaizheng.wang@kuleuven.be

Shireen Kudukkil Manchingal
Oxford Brookes University
skudukkil-manchingal@brookes.ac.uk

Muhammad Mubashar
Oxford Brookes University
mmubashar@brookes.ac.uk

Fabio Cuzzolin
Oxford Brookes University
fabio.cuzzolin@brookes.ac.uk

June 11, 2025

ABSTRACT

Uncertainty estimation is pivotal in machine learning, especially for classification tasks, as it improves the robustness and reliability of models. We introduce a novel 'Epistemic Wrapping' methodology aimed at improving uncertainty estimation in classification. Our approach uses Bayesian Neural Networks (BNNs) as a baseline and transforms their outputs into belief function posteriors, effectively capturing epistemic uncertainty and offering an efficient and general methodology for uncertainty quantification. Comprehensive experiments employing a Bayesian Neural Network (BNN) baseline and an Interval Neural Network for inference on the MNIST, Fashion-MNIST, CIFAR-10 and CIFAR-100 datasets demonstrate that our Epistemic Wrapper significantly enhances generalisation and uncertainty quantification.

1 Introduction

In the realm of machine learning, particularly in classification tasks, uncertainty estimation plays a crucial role in enhancing the robustness and reliability of models [43]. Accurately quantifying uncertainty is vital for applications where decisions must be made with confidence, such as in medical diagnosis [28], autonomous driving [14] and financial forecasting. Traditional deterministic neural networks, while powerful, cannot often effectively capture and express uncertainty [32]. This shortfall has spurred interest in probabilistic approaches, with Bayesian neural networks (BNNs) emerging as a promising solution in this context. BNNs offer a principled approach to uncertainty estimation by incorporating prior distributions over the model parameters, leading to posterior distributions that reflect model uncertainty [23]. Despite their theoretical appeal, BNNs face practical challenges, including high computational costs and complexity in training.

The literature majors on two sources of uncertainty: *Epistemic* Uncertainty (EU) and *Aleatoric* Uncertainty (AU) [22, 1]. Epistemic uncertainty is due to a lack of knowledge about the true model parameters and can be reduced with more data or better models. In contrast, aleatoric uncertainty (AU) stems from the inherent randomness in the data generation process and cannot be reduced. Over the years, various studies [22, 1] have recognised that accurately modelling parameter uncertainty can produce a variety of credible network models, which are likely to include the true underlying network model, leading to both better EU estimation and more reliable inference. In particular, *second-order uncertainty* frameworks (including belief functions [9]) can be employed to model both EU and AU, effectively expressing 'uncertainty about a prediction's uncertainty' [22, 43].

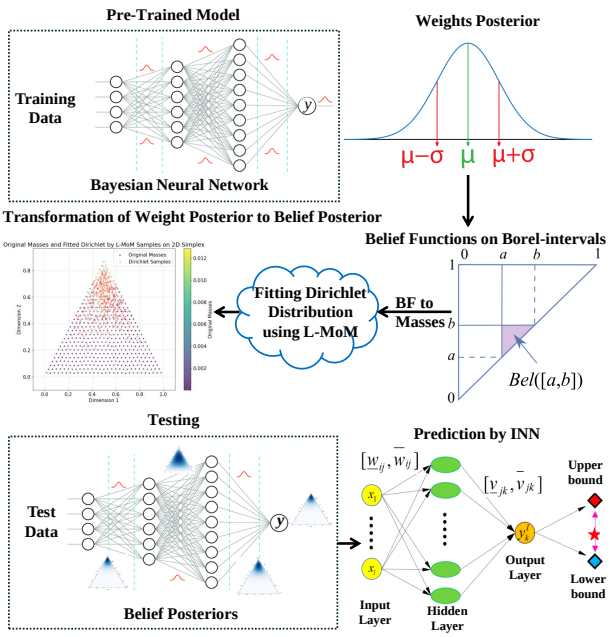


Figure 1: Epistemic Wrapper transforms weights posteriors from a Bayesian Neural Network into belief posteriors through a five-step process. It involves extracting probability posteriors, calculating belief values over Borel intervals, computing mass values using Moebius inversion, fitting a Dirichlet distribution to these masses via method of L-moments, and using the resulting belief posteriors as weights to Interval Neural Networks (INNs) for final predictions.

BNNs, as one of the prevalent method for uncertainty estimation, treat all the weights and biases of the network as probability distributions. The prediction of the NN is represented as a second-order distribution, thus representing the probability distribution of distributions [22]. Although effective approximation techniques have been developed, such as variational inference approaches [3, 16] and sampling methods [41, 19], the high computational cost of BNNs during training as well as inference time limit their practical adoption, especially in real-time applications [1].

Recent results support the claim that modelling EU using uncertainty measures more general than probability distributions [11], such as credal sets [30] or random sets / belief functions [45], can lead to better uncertainty estimation and robustness [35, 38, 37, 5, 51, 50, 4, 36]. Still, all those efforts model (epistemic) uncertainty in the model's *target* space, rather than its *parameter* space. To our knowledge, no attempts have been made to model EU in the parameter space via higher-order uncertainty measures.

This paper proposes *Epistemic Wrapper*, a novel method which, for the first time, models EU in the parameter space via a random set representation by "wrapping" a learnt Bayesian posterior there in the form of a *belief function*. posterior (Fig. 1). Our Epistemic Wrapper follows a structured five-step process, where each step is executed in a hierarchical manner: (i) We begin by extracting posterior distributions with parameters (μ, σ) from a pre-trained BNN model, where the priors are modeled as Gaussian distributions. (ii) The posterior distributions are then truncated, and continuous belief functions are computed over closed intervals. (iii) In the third step, these belief values are transformed into mass values using Moebius inversion. (iv) A Dirichlet distribution is fitted to the grid of mass values using the Method of L-moments. (v) Finally, inference is performed using a Hybrid Interval Neural Network (INN).

Our approach leverages the strengths of BNNs while injecting the ability of higher-order measure to improve robustness and uncertainty estimation.

Our contributions are therefore:

1. **A first attempt to model EU in the parameter space using higher-order uncertainty measures**.
2. This happens via a new, versatile **Epistemic Wrapper** concept, that can be applied to any BNN baseline to convert it automatically into a belief-function posterior.
3. Based on the above, a **novel approach to uncertainty estimation** in classification which efficiently leverages BNNs as a foundation.

Our experiments demonstrate the versatility of the proposed Epistemic-wrapper approach across the BNN baseline on two datasets: MNIST and Fashion-MNIST. The results indicate that the epistemic wrapper generalizes effectively across

these diverse datasets, significantly improving performance over the baseline BNN. On MNIST dataset, the baseline BNN achieved an accuracy of $72.44\% \pm 0.24$, whereas the Epi-Wrapper substantially outperformed it, achieving $91.02\% \pm 0.05$. Similarly, on the Fashion-MNIST dataset, the baseline BNN attained an accuracy of $58.91\% \pm 0.24$, while the Epi-Wrapper demonstrated a notable improvement, reaching $82.45\% \pm 0.10$. These results highlight the effectiveness of our approach in enhancing predictive performance across different datasets.

Why do we model epistemic uncertainty in the parameter space rather than in the target space?
The motivation behind modeling epistemic uncertainty in the parameter space using higher-order uncertainty measures stems from the idea that parameter uncertainty is a primary source of epistemic uncertainty. By modeling uncertainty directly in the parameter space before it propagates to predictions we capture model-level uncertainty in a more principled way. Modeling in the parameter space offers several advantages: (a) It provides a prior-agnostic mechanism to represent epistemic uncertainty, without relying solely on the model's output distribution. (b) It enables structured and interpretable sampling through belief functions and Dirichlet distributions, supporting more stable and calibrated uncertainty estimates via interval-based inference. (c) It can be seamlessly integrated with existing BNNs without retraining, offering flexibility and broader applicability.
The paper is organised as follows. Section 2 surveys the relevant literature. Section 3 explains in detail our epistemic wrapper approach. Section 4 reports the experiments and results. Section 5 concludes.

2 Relevant Work

2.1 Bayesian Neural Networks

Bayesian Neural Networks (BNNs) provide a principled framework for uncertainty estimation by treating weights and biases as probability distributions, supporting robust decision-making [23]. Efficient methods like Variational Inference (e.g., Bayes by Backprop [3]) and dropout-based approximations [16] have made BNNs scalable. However, their high computational demands, especially during training and inference, remain a significant challenge for practical applications [1].

While various types of uncertainty measures [10] have been employed in machine learning in the past [12, 8, 33, 17], recent advancements in epistemic uncertainty modelling have introduced a range of methods to improve predictive reliability across various neural architectures. Evidential deep learning predicts second-order probability distributions to estimate uncertainty, but faces challenges in optimisation and interpretation [24]. Methods like G-ΔUQ refine uncertainty calibration in Graph Neural Networks (GNNs) through stochastic data centering [49], while SPDE-based GNNs employ Q-Wiener processes for uncertainty propagation in complex graphs [31]. The Graph Energy-Based Model (GEBM) leverages graph diffusion to quantify uncertainty at different structural levels [15], and credal set-based ensemble learning constructs plausible probability distributions to measure aleatoric and epistemic uncertainty [20].

Crucially, [36] introduces a unified evaluation framework for uncertainty-aware classifiers, mapping all uncertainty-aware predictions into credal sets [7], thus enabling a standardised assessment of epistemic uncertainty across BNNs, Deep Ensembles, Evidential Deep Learning (EDL), and Credal Set-based approaches. [38] extends uncertainty modelling through Random-Set Neural Networks (RS-NNs), which employ random set theory to construct belief-based uncertainty representations, providing a more flexible alternative to conventional probabilistic models. Credal Interval Neural Networks [52], instead, represent predictions as credal sets, which encapsulate a range of probable outcomes, thereby explicitly modelling epistemic uncertainty. Building on the latter, Credal Deep Ensembles [51] predict and aggregate ensembles of convex sets of probability distributions, resulting in a more conservative and informative epistemic uncertainty quantification. In an alternative approach [6, 34, 44] predictions are modelled as Dirichlet distributions. A key challenge with these methods is the lack of ground truth labels for uncertainty, making direct supervision difficult.

While these models can be highly effective, they primarily quantify uncertainty at the target level, leaving the question of modelling epistemic uncertainty at parameter level open. In contrast, our proposed Epistemic Wrapper leverages BNNs to do exactly so, by transforming probability posteriors into belief posteriors, to offer a robust solution for uncertainty quantification in classification tasks.

2.2 Interval neural networks

Traditional *Interval Neural Networks* (INNs) employ deterministic interval-based representations for inputs, outputs, weights, and biases, ensuring robust uncertainty modelling in neural computations. The forward propagation in an INN follows interval arithmetic principles, where the interval-formed activations in each layer are computed using element-wise interval addition, subtraction, and multiplication [18]. Specifically, the activation output of the l^{th} layer is determined by applying a monotonically increasing activation function to the interval-weighted sum of the previous layer's outputs and the corresponding interval biases. This formulation guarantees the *set constraint* property, ensuring

that for any given input and network parameters within their defined intervals, the computed activations remain bounded within a well-defined range. When the activation function is non-negative (e.g., ReLU), further simplifications allow efficient computation of interval bounds using minimum and maximum operators. This structured interval propagation enables INNs to maintain rigorous mathematical constraints while modelling uncertainties in deep learning architectures [40].

In our approach, we employ INNs at inference time using our epistemic wrapper weights, sampled from the wrapped belief posterior, thus leveraging their structured interval-based representations to quantify and propagate epistemic uncertainty effectively.

3 Methodology

The proposed Epistemic Wrapper approach consists of a five-step process for transforming learnt posterior distributions on model parameters into belief functions posteriors.

3.1 Learning a Bayesian Posterior (Baseline)

In the first step we use BNNs trained on Gaussian priors with parameters (μ, σ) and obtain posterior distributions with the same parameters. The posterior distribution $p(\boldsymbol{\omega}|\mathbb{D})$ is learned using Bayes' theorem, where 'Variational Inference' (VI) is employed to approximate the intractable posterior by optimizing a variational distribution $q(\boldsymbol{\omega})$ that closely matches $p(\boldsymbol{\omega}|\mathbb{D})$. During inference, we approximate the posterior via Bayesian Model Averaging by sampling weights from the variational posterior.

3.2 Dynamic Truncation

A typical Bayesian weight posterior will look like the one in Figure 2.

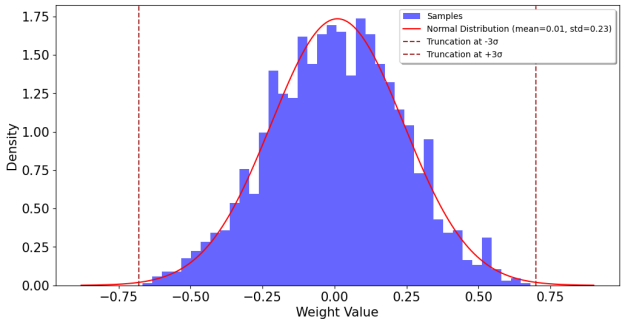


Figure 2: Posterior distribution of weights for the last layer. The histogram displays the sampled weights, overlaid with a fitted normal distribution (mean = 0.01, std = 0.23). Vertical dashed lines indicate truncation points at $\pm 3\sigma$.

In the second step, such posterior distributions are truncated using a *dynamic distribution truncation* mechanism, designed as an adaptive technique used to define the range of a distribution based on its mean and standard deviation. This dynamically scales the bounds to parameter values according to the variance of the distribution, ensuring that the truncation bounds are tighter for distributions with smaller variances, as they inherently have more concentrated probability masses, and looser for those with larger variances.

The truncation bounds are calculated as: $\text{Lower Bound} = \mu - \text{dynamic\_multiplier} \cdot \sigma$, $\text{Upper Bound} = \mu + \text{dynamic\_multiplier} \cdot \sigma$, where μ is the mean, σ is the standard deviation, and the dynamic_multiplier is calculated as: $\text{dynamic\_multiplier} = \min(5.0, \frac{1.0}{\sigma})$, ensuring that the multiplier decreases for low-variance distributions while capping its value at 5.0 to prevent excessive truncation in high-variance cases. We have selected this approach as it provides a balance between capturing the significant probability mass of the distribution and avoiding overly wide or narrow bounds, which could either dilute meaningful mass representation or exclude critical probabilistic regions.

3.3 Continuous Belief Functions on Closed Intervals

Belief Functions. Belief functions, grounded in the mathematical framework of random sets, were initially introduced by Dempster [13] and later formalized by Shafer [46] as an alternative model for subjective belief to Bayesian probability. In finite domains, such as a collection of classes, belief functions are characterised by a *basic probability assignment* (BPA) [46], which is a set function $m : 2^{\Theta} \to [0, 1]$ satisfying $m(\emptyset) = 0$ and $\sum_{A \subseteq \Theta} m(A) = 1$. The value $m(A)$ is

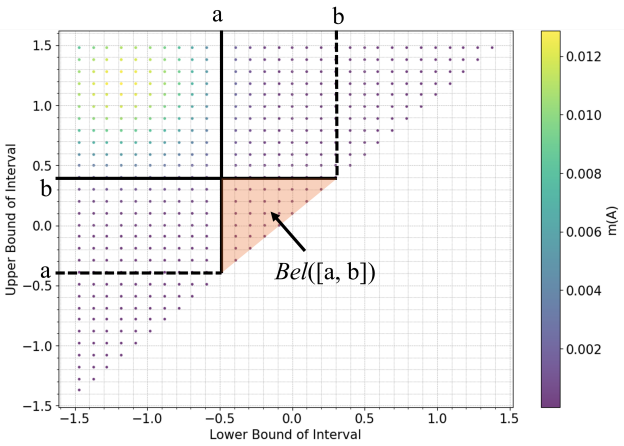


Figure 3: Graphical visualisation of the continuous PDF/mass function on intervals, with the area whose integral amounts to $Bel([a, b])$.

interpreted as the probability mass directly assigned to subset $A \subseteq \Theta$ in a random-set formulation [47]. Subsets A of Θ with $m(A) > 0$ are referred to as *focal elements*. Classical belief functions extend the notion of discrete mass functions by assigning normalized, non-negative mass values not only to elements $\theta \in \Theta$ but to subsets of Θ, governed by:

$$m(A) \geq 0, \forall A \subseteq \Theta, \quad \sum_{A \subseteq \Theta} m(A) = 1. \tag{1}$$

The belief function $Bel(A)$ associated with a mass function m is defined as the total mass assigned to all subsets $B \subseteq A$. Conversely, m can be recovered from Bel through Moebius inversion [46]:

$$Bel(A) = \sum_{B \subseteq A} m(B), \quad m(A) = \sum_{B \subseteq A} (-1)^{|A \setminus B|} Bel(B). \tag{2}$$

This formulation demonstrates that classical probability measures are a special case of belief functions, assigning mass exclusively to singletons.

Continuous belief functions on intervals. Belief functions can be easily extended to continuous spaces (e.g., a network's parameter space) by defining a continuous mass function over the collection of *closed intervals*, rather than the entire power set. Given a network parameter ω with values in $\mathbb{R}$, this requires defining a continuous PDF over the collection of intervals $[a, b] \subset \mathbb{R}$ [9]. Here we will assume that parameter values are bounded after truncation (for illustration, in $[0, 1]$); however, the method can be easily extended to unbounded parameter values as well.

The space of all closed intervals in $[0, 1]$ is a triangle, as illustrated in Fig. 3. Given a continuous mass function there (non-negative and with integral 1), one can compute the belief and plausibility value of a parameter interval $A = [a, b]$ by integrating it over specific regions of the triangle [48] (Fig. 3). The same applies for parameters bounded by arbitrary values.

Given a posterior distribution over a network's weight, learned by a BNN, our Wrapper transforms it into a continuous belief function using the method proposed in [53]. For any closed interval $A = [a, b]$ of the parameter space, one can compute its *plausibility* from the posterior distribution by taking the supremum of the normalised posterior $\hat{p}(\boldsymbol{\omega}|\mathbb{D})$ across all $\boldsymbol{\omega} \in A$, namely:

$$Pl_\Theta(A|\mathbb{D}) = \sup_{\omega \in A} \hat{p}(\boldsymbol{\omega}|\mathbb{D}). \tag{3}$$

The corresponding belief value is then calculated as the complement of the plausibility:

$$Bel_\Theta(A|\mathbb{D}) = 1 - Pl_\Theta(A^c|\mathbb{D}), \tag{4}$$

ultimately providing the sought random-set representation in the parameter space.

The method is grounded into rationality principles, such as (i) the likelihood principle, (ii) compatibility with Bayesian inference (which ensures that combining a Bayesian prior with the belief function yields the Bayesian posterior), and (iii) the principle of Minimum Commitment, which maintains that among the belief functions satisfying the previous two principles, the one chosen should commit to the least amount of information necessary [9].

To cap complexity, sample belief values can be computed for a grid of parameter values only. The corresponding mass values can then be easily obtained by Moebius inversion [45].

3.4 Fitting a Dirichlet Distribution

The fourth step of our Epistemic wrapper employs the method of *L-moments* [21] to fit a Dirichlet distribution to the grid of mass values so obtained.

A **Dirichlet distribution** is a family of continuous multivariate probability distributions parameterised by a vector α of positive real numbers; in fact, a multivariate extension of the Beta distribution (Figure 4):

$$f(x_1, \ldots, x_K; \alpha_1, \ldots, \alpha_K) = \frac{1}{B(\alpha)} \prod_{i=1}^{K} x_i^{\alpha_i - 1}. \tag{5}$$

As they are defined on the collection of vectors $x \in [0,1]^K$ of dimension K whose coordinates add to 1, Dirichlet distributions can be interpreted as second-order distributions.

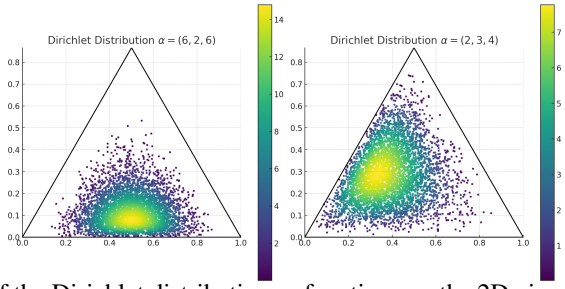


Figure 4: Probability densities of the Dirichlet distribution as functions on the 2D-simplex: α = (6,2,6) (left), α = (2,3,4) (right).

The **method of L-Moments** is a statistical approach employed for parameter estimation in probability distributions. Here we utilise this method to estimate the parameters of a Dirichlet distribution over mass values. L-moments are analogous to conventional moments but are based on linear combinations of order statistics, making them more robust to outliers and capable of providing a more reliable characterization of the data.

To fit a Dirichlet distribution to the grid of mass values, we compute weighted L-moments from the data represented in a 3D simplex space, where each data point has an associated weight derived from its mass value. An example grid in a 3D simplex representation is shown in Figure 5-Left.

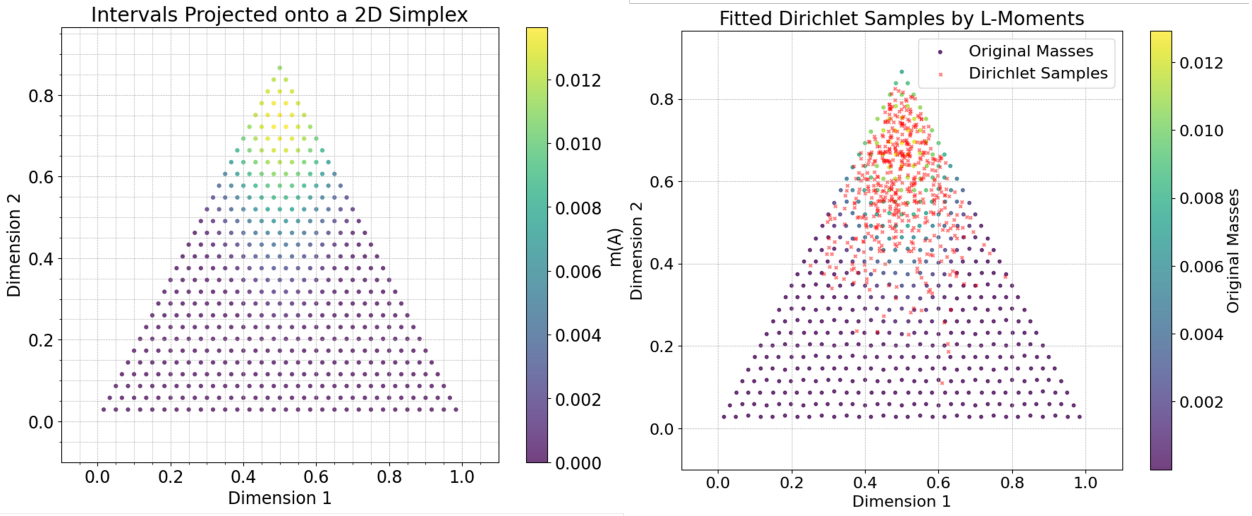


Figure 5: **Left:** Scatter plot showing intervals projected onto a 2D simplex. Each point represents an interval $A = [a, b]$ with its location determined by the values a and b, and the colour scale indicates the corresponding mass values $m(A)$, ranging from 0.00 to 0.012. **Right:** Visualization of Dirichlet samples on a 2D simplex. The scatter plot shows points sampled from the fitted Dirichlet distribution over mass values.

Computation of weighted L-Moments. We first need to compute the first-order and second-order weighted L-moments from the grid of data points. Let $\mathbf{x}_i \in \mathbb{R}^3$ denote the i-th data point in the 3D simplex and w_i its associated weight (derived by normalizing the mass values, so that $\sum_i w_i = 1$). The L-moments are computed as follows. *First-order*

Algorithm 1 Weighted L-Moments for Dirichlet Parameter Estimation

Require: $\mathbf{X} \in \mathbb{R}^{n\times 3}$ (3D simplex coordinates), $\mathbf{w} \in \mathbb{R}^n$ (masses), $\epsilon > 0$
1: Normalize weights: $\mathbf{w} \leftarrow \mathbf{w}/\sum_{i=1}^n w_i$
2: Compute $L_1 \leftarrow \sum_{i=1}^n w_i \mathbf{x}_i$
3: Compute $L_2 \leftarrow \frac{\sum_{i=1}^n w_i(\mathbf{x}_i - L_1)^2}{\sum_{i=1}^n w_i}$
4: Adjust $L_2 \leftarrow \max(L_2, \epsilon)$
5: Estimate $\boldsymbol{\alpha} \leftarrow L_1 \odot \left(\frac{L_1 \odot (1-L_1)}{L_2} - 1\right)$
6: Enforce positivity: $\boldsymbol{\alpha} \leftarrow \max(\boldsymbol{\alpha}, \epsilon)$
Ensure: Dirichlet parameters $\boldsymbol{\alpha}$

L-moment (L_1). This is the weighted mean of the points in the simplex and is given by:

$$L_1 = \sum_{i=1}^{n} w_i \mathbf{x}_i. \tag{6}$$

Second-order L-moment (L_2). This represents the weighted spread (variance) of the points relative to L_1:

$$L_2 = \frac{\sum_{i=1}^n w_i(\mathbf{x}_i - L_1)^2}{\sum_{i=1}^n w_i} \tag{7}$$

To ensure numerical stability, a small value ϵ is added to L_2 when necessary, preventing division by zero in subsequent computations.

Fitting the Dirichlet Distribution. Using the computed L-moments, we can estimate the parameters $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \alpha_3)$ of the Dirichlet distribution. The relationship between L-moments and the Dirichlet parameters is expressed as:

$$\alpha_k = L_{1,k}\left(\frac{L_{1,k}(1 - L_{1,k})}{L_{2,k}} - 1\right), \quad k = 1, 2, 3 \tag{8}$$

where $L_{1,k}$ and $L_{2,k}$ are the respective components of the first and second L-moments along each axis of the simplex. The whole procedure is summarised in Algorithm 1.

As a sanity check, after fitting a Dirichlet distribution to the grid of mass values, samples from it are also concentrated on the top of the simplex as shown in Figure 5-Right.

Theoretical Properties of the Epistemic Wrapper
The Epistemic Wrapper preserves an important theoretical property. Specifically, the original Bayesian posterior P lies within the credal set induced by the belief and plausibility functions after wrapping, satisfying

$$Bel(A) \leq P(A) \leq Pl(A) \quad \text{for all measurable sets } A.$$

This relation ensures that our transformation is conservative: it enriches the original posterior with second-order uncertainty without distorting the underlying predictive information. Consequently, the model maintains consistency with the Bayesian posterior while gaining robustness, which helps to explain the observed improvements in generalization and uncertainty estimation. The plausibility $Pl(A)$ captures the maximum value of $\hat{p}(\omega|\mathbb{D})$ over A, while belief $Bel(A)$ captures the minimum guaranteed mass by considering the complement A^c. Since $P(A)$ is the integral of $\hat{p}(\omega|\mathbb{D})$ over A, it must lie between the least conservative estimate (Bel) and the most generous estimate (Pl) over A. This follows from the construction rules of likelihood-based belief functions and random set theory (see [46, 53, 9]).

3.5 Budgeting

A budgeting strategy is introduced (detailed in Section 4.2) to selectively transform the posterior distributions of a *subset* of parameters (weights and biases). Posteriors that are not selected, referred to as *unwrapped* posteriors, retain their original learned parameters. We propose four distinct budgeting strategies: three are parameter-based, prioritizing posteriors with high μ, high σ, or simultaneously high μ and σ, while the fourth employs a random selection strategy that remains unbiased with respect to these parameter values.

3.6 Inference via Hybrid Interval Neural Networks

The fifth step in our approach is inference. For this purpose, we employ a Hybrid Interval Neural Network *Hybrid-INN*, where weight intervals are derived from a combination of Dirichlet-derived intervals (wrapped parameters) and Gaussian

Table 1: Classification accuracies on MNIST under different budgeting criteria before and after fine-tuning for posterior weights. Results are from 15 runs. Best scores are presented in bold.

BUDGETING	MLP SIZE	BEFORE FINE-TUNING		AFTER FINE-TUNING		
		HYBRID-INN	EPI-WRAPPER	BNN	HYBRID-INN	EPI-WRAPPER
$\uparrow \sigma$	2	9.11 ± 0.53	$\mathbf{12.93 \pm 0.75}$	33.14 ± 0.22	57.77 ± 0.81	$\mathbf{62.43 \pm 0.46}$
	4	10.44 ± 0.77	$\mathbf{19.94 \pm 0.47}$	40.19 ± 0.55	83.32 ± 0.24	$\mathbf{85.17 \pm 0.10}$
	8	9.33 ± 0.54	$\mathbf{25.46 \pm 1.57}$	72.44 ± 0.24	$\mathbf{91.12 \pm 0.08}$	91.08 ± 0.09
$\uparrow \mu$	2	9.11 ± 0.53	$\mathbf{10.63 \pm 0.34}$	33.14 ± 0.22	57.77 ± 0.81	$\mathbf{63.06 \pm 0.47}$
	4	10.44 ± 0.77	$\mathbf{18.13 \pm 0.71}$	40.19 ± 0.55	83.32 ± 0.24	$\mathbf{85.35 \pm 0.06}$
	8	9.33 ± 0.54	$\mathbf{51.33 \pm 1.21}$	72.44 ± 0.24	$\mathbf{91.12 \pm 0.08}$	91.02 ± 0.05
$\uparrow \mu + \sigma$	2	9.11 ± 0.53	$\mathbf{10.45 \pm 0.16}$	33.14 ± 0.22	57.77 ± 0.81	$\mathbf{63.02 \pm 0.55}$
	4	10.44 ± 0.77	$\mathbf{18.55 \pm 0.68}$	40.19 ± 0.55	83.32 ± 0.24	$\mathbf{85.18 \pm 0.07}$
	8	9.33 ± 0.54	$\mathbf{51.31 \pm 1.29}$	72.44 ± 0.24	91.12 ± 0.08	$\mathbf{91.12 \pm 0.07}$
RS	2	$\mathbf{9.11 \pm 0.53}$	9.80 ± 0.00	33.14 ± 0.22	57.77 ± 0.81	$\mathbf{64.84 \pm 0.16}$
	4	10.44 ± 0.77	$\mathbf{17.35 \pm 0.27}$	40.19 ± 0.55	83.32 ± 0.24	$\mathbf{85.45 \pm 0.06}$
	8	$\mathbf{9.33 \pm 0.54}$	9.23 ± 0.64	72.44 ± 0.24	$\mathbf{91.12 \pm 0.08}$	90.80 ± 0.09

posteriors (unwrapped parameters). Although our architecture is based on a standard INN, the way we handle these intervals is by computing the mean of the upper and lower bounds, which introduces an important distinction. For this reason, we refer to our model as Hybrid-INN. This averaging mechanism of the interval bounds prevents extreme values from distorting predictions. It ensures a more stable, well-calibrated uncertainty representation, particularly for epistemic uncertainty, which can otherwise be highly sensitive to interval width. It also harmonizes wrapped and unwrapped weights by smoothly integrating both Dirichlet and Gaussian-based uncertainty representations.

Namely, the unwrapped weights (Gaussian posteriors) generate the following intervals: $\text{Lower bound} = \mu - \sigma, \text{Upper bound} = \mu + \sigma$. The process of defining these wrapped and unwrapped weights can be considered the weight initialisation step for the Hybrid-INN model. In contrast, the baseline Hybrid-INN retains the default weight initialisation scheme [25]. In other words, at inference time, the model operates as a Hybrid-INN but with two distinct sets of weights: (i) with random initialisation, (baseline Hybrid-INN) and (ii) with weights transformed by the Epi-Wrapper. Both models are also fine-tuned: the baseline with randomly initialized weights, while the Epi-Wrapper utilizes weights transformed through the wrapping strategy. For a fair comparison, both the baseline and our Epi-Wrapper operate under identical functional settings.

4 Experiments

In this section, we present a comprehensive evaluation of our Epi-Wrapper approach through a series of experiments designed to assess its effectiveness in uncertainty estimation and predictive performance. We begin by describing the experimental setup, including datasets, model architecture, and ablation studies. We then compare our method against relevant baselines, followed by an analysis of key performance metrics.

4.1 Implementation Details

Datasets. We evaluated the performance of the Epistemic Wrapper on three classification benchmarks: MNIST [29], Fashion-MNIST [55] and CIFAR-10 [27]. The **MNIST** dataset comprises 70,000 grayscale images of handwritten digits (0–9), each with a resolution of 28×28 pixels, and is mostly used for classification and pattern recognition tasks due to its simplicity and accessibility. **Fashion MNIST** serves as a more challenging alternative to MNIST, containing 70,000 grayscale images of fashion items, such as shirts, shoes, and bags, also at same resolution of 28×28 pixels. This dataset provides a greater diversity in texture and structure, making it suitable for evaluating model's generalization capabilities.

As **baseline** we use a standard variational BNN [2], applied to the classical Multilayer Perceptron (MLP) architecture. The **MLP Backbone** is composed of an input layer, a single hidden layer and an output layer. The **Input Layer** processes the input data with a shape corresponding to the dimensions of the dataset. For grayscale datasets (MNIST and Fashion MNIST), the input shape is $28 \times 28 \times 1$, and for CIFAR-10, the input shape is $32 \times 32 \times 3$. A **Flattening Layer** flattens the input into a single-dimensional vector to be fed to the subsequent dense layers. **DenseFlipout Layers** are implemented using TensorFlow Probability's `DenseFlipout`. They approximate the weight posterior distributions using a Flipout Monte Carlo estimator, which reduces the variance of gradient estimates during backpropagation. The first dense layer contains hidden units with ReLU activation, followed by a dropout layer with a rate of 0.1 to

prevent overfitting. The second dense layer, which acts as the output layer, maps to the number of classes in the dataset. The Bayesian MLP is trained using the Evidence Lower Bound (ELBO) loss function, which combines the negative log-likelihood (NLL) of the observed data with the Kullback-Leibler (KL) divergence between the approximate posterior and the prior distributions of the weights. The NLL component is computed using the softmax cross-entropy between the true labels and the predicted logits, while the KL divergence is derived from the prior and posterior distributions of the weights. The model is trained over 20 epochs (for MNIST and Fashion MNIST) and using Adam.

A number of hyperparameters are fixed across all experiments: namely, the number of closed intervals (30) and the number of samples drawn from the posterior distributions (5,000). Additionally, the budgeting strategy is consistently applied by selecting 5% of the weights based on the selected criteria specified per experiment.

4.2 Ablation on Budgeting

We first conducted an ablation study on the MNIST dataset in which four different Budgeting criterias were tested.

In **Budgeting using High Variance ($\uparrow \sigma$)** we sampled 5% weights with 'High Variance' from the posterior distributions (parameters: μ, σ) of the whole model and transformed them to belief posteriors using Epistemic Wrapper. The results are shown in Table 1, where "MLP size" is the number of hidden units in the single hidden layer of the model.

Since inference in our methodology is done using INNs, we compare our results with those of Hybrid INN (taken as a baseline). The results shows that using the wrapper improves the quality of the weights initialization with respect to the Hybrid INN baseline. For instance, an MLP with 32 hidden units and weights randomly initialized achieved an accuracy of 10.37% on the test data, while for our wrapper the test accuracy was 50.20%.

Budgeting using High Mean ($\uparrow \mu$) is another strategy in which we sample and "wrap" the 5% weights with 'High Mean' from the posterior distributions. From the results shown in Table 1, it can be seen that "High Mean" performs better for MLP size (no hidden units) $= 8$.

In Budgeting using High Mean and High Variance ($\uparrow (\mu, \sigma)$) we rank the parameters by computing a combined score, defined as the sum of the mean and variance of their posterior distributions: combined_score $= \mu + \sigma$. This acts as a proxy for an upper bound of the posterior distribution, allowing us to prioritize parameters that are either highly informative (high mean) or uncertain (high variance). We then wrap these top 5% weights using Epi-wrapper. The results are shown in the Table 1. This strategy allows us to selectively wrap the most influential and uncertain parameters, ensuring that the transformation captures meaningful epistemic uncertainty. However, this approach also imposes a strict constraint on the selection process, as only weights satisfying both conditions are chosen, which may limit flexibility in certain scenarios.

Table 2: Classification accuracies on MNIST and Fashion-MNIST datasets for posterior weights transformations using 'High Mean' Budgeting (5% posterior distributions selection) before and after fine-tuning. Results are from 15 runs. MLP (Hidden Units = 8).

DATASETS	BEFORE FINE-TUNING		AFTER FINE-TUNING		
	HYBRID-INN	EPI-WRAPPER	BNN	HYBRID-INN	EPI-WRAPPER
MNIST	9.33 ± 0.54	$\mathbf{51.33 \pm 1.21}$	72.44 ± 0.24	$\mathbf{91.12 \pm 0.08}$	91.02 ± 0.05
FASHION-MNIST	8.57 ± 1.03	$\mathbf{26.93 \pm 1.44}$	58.91 ± 0.24	82.41 ± 0.19	$\mathbf{82.45 \pm 0.10}$

Budgeting using Random Selection (RS) (μ, σ) is done by randomly selecting 5% weights from the baseline BNN and extract belief posteriors using the wrapper. Table 1 shows that the results are worse than with other strategies. This is due to the fact that random sampling, while giving us an unbiased selection of posterior weights, may miss those posterior distributions with high uncertainty that can be improved using our wrapping approach.

4.3 Fine-Tuning

We performed the Fine-tuning of the models, the baseline (Hybrid-INN) and ours (Epi-Wrapper), on the training data. The results are shown in Table 1. In comparison to the Hybrid-INN and BNN baselines, our model performs well as the weights wrapping mechanism acts as an initialization strategy in fine-tuning.

4.4 iD and OoD Experimental Evaluation

Analysis of Accuracy Rejection Curve (ARC) on in-Domain Data Figure 6 presents the Accuracy Rejection Curve (ARC) for the in-domain (iD) dataset, comparing the performance of two models, Hybrid-INN and Epi-Wrapper,

Table 3: Performance comparison regarding uncertainty estimation on MNIST vs Fashion-MNIST OoD samples.

	iD Test Accuracy (%)	OoD Samples		
		AUROC	AUPRC	EU
Hybrid-INN	91.07 ± 0.08	0.5329	0.8951	0.0023
Epi-Wrapper	$\mathbf{91.12 \pm 0.06}$	**0.6673**	**0.9120**	**0.0059**

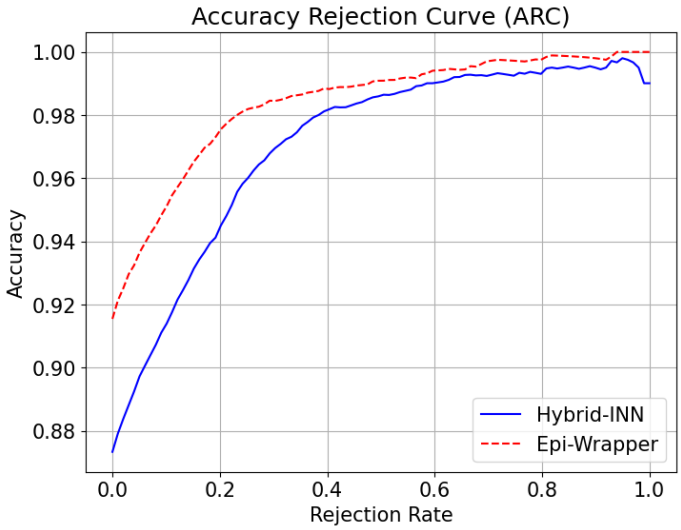


Figure 6: Accuracy Rejection Curve (ARC) for the MNIST dataset, comparing Hybrid-INN and Epi-Wrapper.

on MNIST. In the in-domain setting, both the training and testing are conducted on the MNIST dataset, with no distributional shift. It can be seen in the plot that, as the rejection rate increases, the accuracy steadily improves for both models, demonstrating that removing high-uncertainty predictions enhances the overall correctness of classification. However, the Epi-Wrapper model consistently outperforms the Hybrid-INN model across the entire range of rejection rates. This suggests that Epi-Wrapper is better at identifying and filtering uncertain predictions, leading to an increase in accuracy. Since this evaluation is conducted in an in-domain setting (iD), the observed performance improvements primarily reflect how well the models handle uncertainty within a familiar data distribution. For OOD samples, the results are presented in Table 3. Clearly our Epi-wrapper performs well in comparison to the baseline (Hybrid-INN).

Uncertainty Evaluation: In this work, we employ Monte Carlo (MC) Dropout to estimate predictive uncertainty by performing multiple stochastic forward passes through the model at inference time. Unlike standard deterministic models, where a single forward pass generates fixed predictions, MC Dropout enables the model to capture both aleatoric and epistemic uncertainty. The results are presented in Table 3.

Table 3 presents a comparative analysis of uncertainty estimation performance between Hybrid-INN and Epi-Wrapper on out-of-distribution (OoD) samples from the Fashion-MNIST dataset, while both models were trained and evaluated in-domain (iD) on MNIST. The table provides key evaluation metrics, including iD test accuracy, AUROC (Area Under the Receiver Operating Characteristic Curve), AUPRC (Area Under the Precision-Recall Curve), and Epistemic Uncertainty. For iD Test Accuracy: Both models achieve comparable in-domain classification accuracy on MNIST, with Hybrid-INN attaining an accuracy of 91.07% and Epi-Wrapper slightly outperforming it with 91.12%. AUROC on OoD Samples: This is a key metric for OoD detection, measuring the model's ability to distinguish between iD and OoD samples. A higher AUROC indicates better separation. Epi-Wrapper achieves a higher AUROC of 0.6673, outperforming Hybrid-INN (0.5329). This suggests that Epi-Wrapper is more effective at recognizing and distinguishing OoD samples from in-domain data. AUPRC on OoD Samples: AUPRC provides insight into how well the model prioritizes high-confidence predictions for OoD detection. Epi-Wrapper achieves an AUPRC of 0.9120, surpassing Hybrid-INN (0.8951). The higher AUPRC indicates that Epi-Wrapper generates better-calibrated uncertainty estimates, ensuring that truly OoD samples receive higher uncertainty scores.

EU represents the model's epistemic uncertainty measure, where higher values suggest better sensitivity to OoD data. Epi-Wrapper achieves an EU of 0.0059, which is more than twice the EU of Hybrid-INN (0.0023). This implies that Epi-Wrapper assigns greater uncertainty to OoD samples, making it more reliable in real-world scenarios where detecting unfamiliar inputs is crucial.

4.5 Comparative Analysis

Table 2 offers a comparative analysis of classification accuracies on the MNIST and Fashion-MNIST datasets. Epi-Wrapper exhibits distinct performance advantages over its counterparts, BNN and Hybrid-INN. Before fine-tuning, Epi-Wrapper significantly outperforms Hybrid-INN on both datasets, achieving high accuracy of 51.33% on MNIST and 26.93% on Fashion-MNIST. This indicates our Wrapper's superior ability to perform better on challenging data. After the fine-tuning process, while the BNN and Hybrid-INN models show notable improvements demonstrating their adaptability through training, the Epi-Wrapper maintains competitive, high-performance levels. Its superiority on the Bayesian baseline, both before and after fine tuning, confirms our hypothesis that employing higher-order, random-set representations in the parameter space is advantageous.

5 CONCLUSIONS

This paper presented a novel methodology, Epistemic Wrapper, which, for the first time, extends higher-order uncertainty representation to the parameter space of neural networks. Utilizing Bayesian neural networks as a baseline, our approach transforms their outputs into belief-function posteriors. This method effectively captures epistemic uncertainty, thus offering a robust, efficient, and generic approach to uncertainty quantification. Our experimental analysis on a BNN baseline with MLP architecture across two datasets, MNIST and Fahion-MNIST, validated the effectiveness of Epistemic Wrappers. The results demonstrated that Epi-Wrapper does generalise well. Our future work is to extend and validate the approach on larger scale networks, and develop a framework to use the wrapped weights to generate a predictive random set in the target space.

References

[1] Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U Rajendra Acharya, et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 76:243–297, 2021.

[2] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

[3] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *Proceedings of the International Conference on Machine Learning*, pages 1613–1622. PMLR, 2015.

[4] Michele Caprio, Maryam Sultana, Eleni Elia, and Fabio Cuzzolin. Credal learning theory. *arXiv preprint arXiv:2402.00957*, 2024.

[5] Matthew Albert Chan, Maria J. Molina, and Christopher Metzler. Estimating epistemic and aleatoric uncertainty with a single model. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[6] Bertrand Charpentier, Daniel Zügner, and Stephan Günnemann. Posterior network: Uncertainty estimation without ood samples via density-based pseudo-counts. *Advances in neural information processing systems*, 33:1356–1367, 2020.

[7] Fabio Cuzzolin. On the credal structure of consistent probabilities. In *European Workshop on Logics in Artificial Intelligence*, pages 126–139. Springer, 2008.

[8] Fabio Cuzzolin. Generalised max entropy classifiers. In Sébastien Destercke, Thierry Denœux, Fabio Cuzzolin, and Arnaud Martin, editors, *Belief Functions: Theory and Applications*, pages 39–47, Cham, 2018. Springer International Publishing.

[9] Fabio Cuzzolin. *The geometry of uncertainty: The geometry of imprecise probabilities*. Springer Nature, 2020.

[10] Fabio Cuzzolin. Uncertainty measures: The big picture. *arXiv preprint arXiv:2104.06839*, 2021.

[11] Fabio Cuzzolin. Uncertainty measures: A critical survey. *Information Fusion*, page 102609, 2024.

[12] Fabio Cuzzolin and Wenjua Gong. Belief modeling regression for pose estimation. In *Proceedings of the 16th International Conference on Information Fusion*, pages 1398–1405, 2013.

[13] Arthur P Dempster. Upper and lower probabilities induced by a multivalued mapping. In *Classic works of the Dempster-Shafer theory of belief functions*, pages 57–72. Springer, 2008.

[14] Stanislav Fort and Stanislaw Jastrzebski. Large scale structure of neural network loss landscapes. *Advances in Neural Information Processing Systems*, 32, 2019.

[15] Dominik Fuchsgruber, Tom Wollschläger, and Stephan Günnemann. Energy-based epistemic uncertainty for graph neural networks. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[16] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the International Conference on Machine Learning*, pages 1050–1059. PMLR, 2016.

[17] Wenjuan Gong and Fabio Cuzzolin. A belief-theoretical approach to example-based pose estimation. *IEEE Transactions on Fuzzy Systems*, 26(2):598–611, 2017.

[18] Timothy Hickey, Qun Ju, and Maarten H Van Emden. Interval arithmetic: From principles to implementation. *Journal of the ACM (JACM)*, 48(5):1038–1068, 2001.

[19] Matthew D Hoffman, Andrew Gelman, et al. The No-U-Turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *J. Mach. Learn. Res.*, 15(1):1593–1623, 2014.

[20] Paul Hofman, Yusuf Sale, and Eyke Hüllermeier. Quantifying aleatoric and epistemic uncertainty: A credal approach. In *ICML 2024 Workshop on Structured Probabilistic Inference $\{\backslash\&\}$ Generative Modeling*, 2024.

[21] J. R. M. Hosking. L-moments: Analysis and estimation of distributions using linear combinations of order statistics. *Journal of the Royal Statistical Society: Series B (Methodological)*, 52(1):105–124, 12 2018.

[22] Eyke Hüllermeier and Willem Waegeman. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning*, 110(3):457–506, 2021.

[23] Laurent Valentin Jospin, Hamid Laga, Farid Boussaid, Wray Buntine, and Mohammed Bennamoun. Hands-on Bayesian neural networks—A tutorial for deep learning users. *IEEE Computational Intelligence Magazine*, 17(2):29–48, 2022.

[24] Mira Juergens, Nis Meinert, Viktor Bengs, Eyke Hüllermeier, and Willem Waegeman. Is epistemic uncertainty faithfully represented by evidential deep learning methods? In *Forty-first International Conference on Machine Learning*, 2024.

[25] LS Kim. Understanding the difficulty of training deep feedforward neural networks xavier. In *Proceedings of the International Joint Conference on Neural Networks*, volume 2, 1993.

[26] Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[27] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. CIFAR-10 (Canadian Institute For Advanced Research). 2009.

[28] Antonis Lambrou, Harris Papadopoulos, and Alex Gammerman. Reliable confidence measures for medical diagnosis with evolutionary algorithms. *IEEE Transactions on Information Technology in Biomedicine*, 15(1):93–99, 2010.

[29] Yann LeCun. The MNIST database of handwritten digits. *http://yann.lecun.com/exdb/mnist/*, 1998.

[30] Isaac Levi. *The enterprise of knowledge: An essay on knowledge, credal probability, and chance*. MIT press, 1980.

[31] Xixun Lin, Wenxiao Zhang, Fengzhao Shi, Chuan Zhou, Lixin Zou, Xiangyu Zhao, Dawei Yin, Shirui Pan, and Yanan Cao. Graph neural stochastic diffusion for estimating uncertainty in node classification. In *Forty-first International Conference on Machine Learning*, 2024.

[32] Jeremiah Liu, Zi Lin, Shreyas Padhy, Dustin Tran, Tania Bedrax Weiss, and Balaji Lakshminarayanan. Simple and principled uncertainty estimation with deterministic deep learning via distance awareness. *Advances in Neural Information Processing Systems*, 33:7498–7512, 2020.

[33] Zhun-Ga Liu, Yu Liu, Jean Dezert, and Fabio Cuzzolin. Evidence combination based on credal belief redistribution for pattern classification. *IEEE Transactions on Fuzzy Systems*, 28(4):618–631, 2019.

[34] Andrey Malinin, Bruno Mlodozeniec, and Mark Gales. Ensemble distribution distillation. *arXiv preprint arXiv:1905.00076*, 2019.

[35] Shireen Kudukkil Manchingal and Fabio Cuzzolin. Epistemic deep learning. *arXiv preprint arXiv:2206.07609*, 2022.

[36] Shireen Kudukkil Manchingal, Muhammad Mubashar, Kaizheng Wang, and Fabio Cuzzolin. A unified evaluation framework for epistemic predictions, 2025.

[37] Shireen Kudukkil Manchingal, Muhammad Mubashar, Kaizheng Wang, Keivan Shariatmadar, and Fabio Cuzzolin. Random-set convolutional neural network (rs-cnn) for epistemic deep learning. *arXiv preprint arXiv:2307.05772*, 2023.

[38] Shireen Kudukkil Manchingal, Muhammad Mubashar, Kaizheng Wang, Keivan Shariatmadar, and Fabio Cuzzolin. Random-set neural networks. In *The Thirteenth International Conference on Learning Representations*, 2025.

[39] Dmitry Molchanov, Arsenii Ashukha, and Dmitry Vetrov. Variational dropout sparsifies deep neural networks. In *Proceedings of the International Conference on Machine Learning*, pages 2498–2507. PMLR, 2017.

[40] Giorgio Morales and John Sheppard. Adaptive sampling to reduce epistemic uncertainty using prediction interval-generation neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, Feb. 2025.

[41] Radford M Neal et al. MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2(11):2, 2011.

[42] Kazuki Osawa, Siddharth Swaroop, Anirudh Jain, Runa Eschenhagen, Richard E. Turner, Rio Yokota, and Mohammad Emtiyaz Khan. Practical deep learning with bayesian principles. *arXiv preprint arXiv:1906.02506*, 2019.

[43] Yusuf Sale, Michele Caprio, and Eyke Höllermeier. Is the volume of a credal set a good measure for epistemic uncertainty? In *Uncertainty in Artificial Intelligence*, pages 1795–1804. PMLR, 2023.

[44] Murat Sensoy, Lance Kaplan, and Melih Kandemir. Evidential deep learning to quantify classification uncertainty. *Advances in neural information processing systems*, 31, 2018.

[45] Glenn Shafer. *A mathematical theory of evidence*, volume 42. Princeton university press, 1976.

[46] Glenn Shafer. *A mathematical theory of evidence*, volume 42. Princeton university press, 1976.

[47] Philippe Smets. The transferable belief model and other interpretations of Dempster–Shafer's model. In P. P. Bonissone, M. Henrion, L. N. Kanal, and J. F. Lemmer, editors, *Uncertainty in Artificial Intelligence*, volume 6, pages 375–383. North-Holland, Amsterdam, 1991.

[48] Philippe Smets. Belief functions on real numbers. *International Journal of Approximate Reasoning*, 40(3):181–223, 2005.

[49] Puja Trivedi, Mark Heimann, Rushil Anirudh, Danai Koutra, and Jayaraman J. Thiagarajan. Accurate and scalable estimation of epistemic uncertainty for graph neural networks. In *The Twelfth International Conference on Learning Representations*, 2024.

[50] Kaizheng Wang, Fabio Cuzzolin, Keivan Shariatmadar, David Moens, and Hans Hallez. Credal wrapper of model averaging for uncertainty estimation on out-of-distribution detection. *arXiv preprint arXiv:2405.15047*, 2024.

[51] Kaizheng Wang, Fabio Cuzzolin, Keivan Shariatmadar, David Moens, Hans Hallez, et al. Credal deep ensembles for uncertainty quantification. *Advances in Neural Information Processing Systems*, 37:79540–79572, 2024.

[52] Kaizheng Wang, Keivan Shariatmadar, Shireen Kudukkil Manchingal, Fabio Cuzzolin, David Moens, and Hans Hallez. Creinns: Credal-set interval neural networks for uncertainty estimation in classification tasks. *Neural Networks*, page 107198, 2025.

[53] Larry A. Wasserman. Belief functions and statistical inference. *Canadian Journal of Statistics*, 18(3):183–196, 1990.

[54] Yeming Wen, Paul Vicol, Jimmy Ba, Dustin Tran, and Roger Grosse. Flipout: Efficient pseudo-independent weight perturbations on mini-batches. In *Proceedings of the International Conference on Learning Representations*, 2018.

[55] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

A Appendix

A.1 Interval Neural Networks (INNs)

Traditional *interval neural networks* use deterministic interval-based inputs, outputs, and parameters (weights and biases) for each node. The forward propagation in the l^{th} layer of INNs is expressed as:

$$\begin{aligned}[\underline{\boldsymbol{a}}, \overline{\boldsymbol{a}}]^l &= \sigma^l([\underline{\boldsymbol{\omega}}, \overline{\boldsymbol{\omega}}]^l \odot [\underline{\boldsymbol{a}}, \overline{\boldsymbol{a}}]^{l-1} \oplus [\underline{\boldsymbol{b}}, \overline{\boldsymbol{b}}]^l) \\ &= [\sigma^l(\underline{\boldsymbol{o}} + \underline{\boldsymbol{b}}), \sigma^l(\overline{\boldsymbol{o}} + \overline{\boldsymbol{b}})] \text{ with} \\ [\underline{\boldsymbol{o}}, \overline{\boldsymbol{o}}]^l &= [\underline{\boldsymbol{\omega}}, \overline{\boldsymbol{\omega}}]^l \odot [\underline{\boldsymbol{a}}, \overline{\boldsymbol{a}}]^{l-1},\end{aligned} \tag{9}$$

where $\oplus$, $\ominus$, and $\odot$ represent interval addition, subtraction, and multiplication, respectively [18]. The terms $[\underline{\boldsymbol{a}}, \overline{\boldsymbol{a}}]^l$, $[\underline{\boldsymbol{a}}, \overline{\boldsymbol{a}}]^{l-1}$, $[\underline{\boldsymbol{\omega}}, \overline{\boldsymbol{\omega}}]^l$, and $[\underline{\boldsymbol{b}}, \overline{\boldsymbol{b}}]^l$ denote the interval-formed outputs of the l^{th} and $(l-1)^{th}$ layers, as well as the intervals of weights and biases of the l^{th} layer, respectively. $\sigma^l(\cdot)$ is the activation function of the l^{th} layer, which must be monotonically increasing. The application of interval arithmetic [18] in (9) grants INNs the 'set constraint' property. Specifically, for any $\boldsymbol{a}^{l-1} \in [\underline{\boldsymbol{a}}, \overline{\boldsymbol{a}}]^{l-1}$, $\boldsymbol{\omega}^l \in [\underline{\boldsymbol{\omega}}, \overline{\boldsymbol{\omega}}]^l$, and $\boldsymbol{b}^l \in [\underline{\boldsymbol{b}}, \overline{\boldsymbol{b}}]^l$, the constraint in (10) consistently holds.

$$\boldsymbol{a}^l = \sigma^l(\boldsymbol{\omega}^l \cdot \boldsymbol{a}^{l-1} + \boldsymbol{b}^l) \in [\underline{\boldsymbol{a}}, \overline{\boldsymbol{a}}]^l. \tag{10}$$

If $[\underline{\boldsymbol{a}}, \overline{\boldsymbol{a}}]$ is non-negative, such as the output of RELU activation, the calculation of $[\underline{\boldsymbol{o}}, \overline{\boldsymbol{o}}]$ in (9) can be simplified as:

$$\begin{aligned}\underline{\boldsymbol{o}} &= \min\{\underline{\boldsymbol{\omega}}, \mathbf{0}\} \cdot \overline{\boldsymbol{a}} + \max\{\underline{\boldsymbol{\omega}}, \mathbf{0}\} \cdot \underline{\boldsymbol{a}} \\ \overline{\boldsymbol{o}} &= \max\{\overline{\boldsymbol{\omega}}, \mathbf{0}\} \cdot \overline{\boldsymbol{a}} + \min\{\overline{\boldsymbol{\omega}}, \mathbf{0}\} \cdot \underline{\boldsymbol{a}}.\end{aligned} \tag{11}$$

A.2 Datasets:

CIFAR-10 is a collection of 60,000 color images (split into 50,000 training and 10,000 testing samples) across 10 classes, including animals and vehicles, with each image having a resolution of 32×32 pixels.
CIFAR-100 consists of 60,000 color images, each of size 32×32 pixels with three RGB channels, divided into 50,000 training images and 10,000 test images. The dataset contains 100 fine-grained classes, with each class having 600 samples, making it a more challenging extension of the CIFAR-10 dataset. Unlike CIFAR-10, which includes only 10 broad categories, CIFAR-100 introduces a hierarchical structure, grouping its 100 classes into 20 superclasses based on semantic similarity.

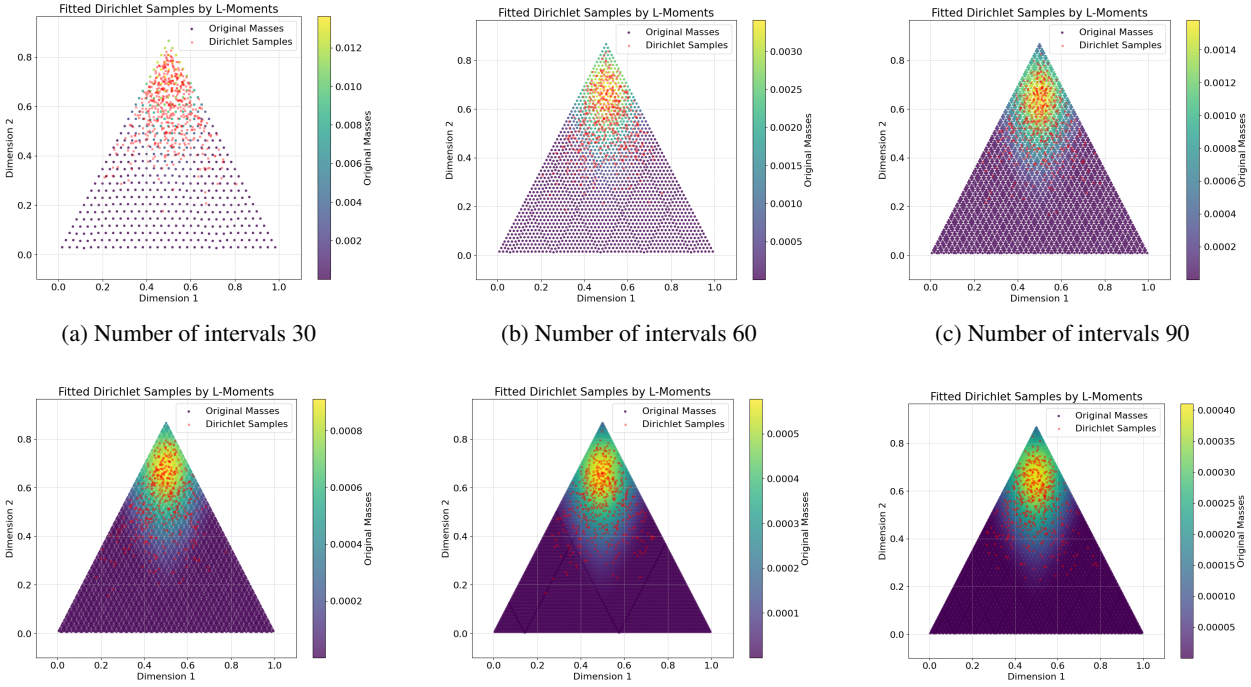
A.3 Bayesian Baselines

For baselines, we utilize two standard variational BNNs: BNNR (Auto-Encoding Variational Bayes [26] with the local re-parameterization trick [39]), and BNNF (Flipout gradient estimator with the negative evidence lower bound loss [54]). The results before fine-tuning are presented in Table 6.

A.4 Backbones

LeNet-5 is adapted in this study into a Bayesian framework. Following are the details of the model's architecture. **Input Layer:** The model accepts input data with a shape corresponding to the dataset used (e.g., $28 \times 28 \times 1$ for grayscale datasets like MNIST and Fashion MNIST, and $32 \times 32 \times 3$ for RGB datasets like CIFAR-10). **Bayesian Convolutional Layers:** The first convolutional layer employs a `Convolution2DFlipout` layer with 6 filters, a kernel size of 5×5, and ReLU activation. This is followed by an average pooling layer to reduce spatial dimensions. The second convolutional layer also uses a `Convolution2DFlipout` layer with 16 filters, a kernel size of 5×5, and ReLU activation. This layer is also followed by an average pooling layer for further downsampling. **Flattening Layer:** After the convolutional layers, the output is flattened into a one-dimensional vector, preparing it for fully connected layers. **Bayesian Fully Connected Layers:** The first fully connected Bayesian layer uses a `DenseFlipout` layer with 120 units and ReLU activation. The second fully connected Bayesian layer uses a `DenseFlipout` layer with 84 units and ReLU activation. **Output Layer:** The output is produced by a `DenseFlipout` layer with a number of units equal to the number of classes in the dataset, applying a softmax activation function to generate class probabilities.

ResNet-18 The Bayesian ResNet-18 model integrates Bayesian inference into the classical ResNet-18 architecture. This model leverages Bayesian Convolutional Neural Networks (Bayesian CNNs) with Flipout and Reparameterization layers from TensorFlow Probability, enabling weight uncertainty modeling. The architecture consists of four main residual blocks, with convolutional layers followed by batch normalization and ReLU activation. The convolutional



(a) Number of intervals 30 (b) Number of intervals 60 (c) Number of intervals 90

(d) Number of intervals 120 (e) Number of intervals 150 (f) Number of intervals 180

Figure 7: Varying Number of closed intervals

layers employ Bayesian weight posterior distributions, where the kernel weights follow a Gaussian posterior parameterized by mean and variance. These distributions are constrained using a log-variance regularization technique, ensuring numerical stability. The weight posteriors are sampled using the Mean-Field Variational Inference approach, enabling Bayesian updates during training. The ResNet-18 backbone begins with an initial convolutional layer followed by four residual blocks, each progressively increasing the number of filters from 64 to 512. The residual connections allow gradient flow through the network, ensuring stable training. To approximate the posterior over weights, Convolution2DReparameterization and Convolution2DFlipout layers are utilized, capturing epistemic uncertainty through stochastic weight sampling. The final layers include average pooling, flattening, and a fully connected Bayesian dense layer with Flipout, producing the classification logits.

VGG-16 The Bayesian VGG-16 model integrates Bayesian inference into the classical VGG-16 architecture to enable principled uncertainty estimation in deep learning. The standard convolutional layers are replaced with Bayesian Convolutional Neural Networks (Bayesian CNNs) using Convolution2DReparameterization and Convolution2DFlipout layers from TensorFlow Probability. These layers approximate posterior distributions over weights using Mean-Field Variational Inference, ensuring reliable uncertainty quantification. VGG-16 follows a deep convolutional architecture with 16 layers, consisting of multiple stacked convolutional layers with small 3×3 filters, followed by max pooling layers to progressively reduce spatial dimensions. The Bayesian adaptation maintains this structure while introducing posterior weight sampling in convolutional layers, ensuring that the feature extraction process incorporates uncertainty information. Batch normalization and ReLU activation are applied to enhance convergence stability, while Bayesian priors constrain weight posteriors, preventing overconfidence in predictions. The final classification layers include Bayesian fully connected layers with Flipout, which sample weights during inference to produce uncertainty-aware predictions.

A.5 Hyperparameter Analysis

The main manuscript experimental setup contains fixed hyperparameters such as number of closed intervals (30) and samples drawn from the posterior distributions (5,000). Also, the budgeting strategy is consistently applied by selecting 5% of the weights based on the selected criteria specified per experiment.

Table 4: Before Fine-tuning: Performance comparison regarding Budgeting percentage on MNIST. Number of intervals fixed 30.

	Epi-Wrapper					
Hybrid INN Accuracy (%)	5% (Weights)	10% (Weights)	20% (Weights)	30% (Weights)	40% (Weights)	50% (Weights)
9.33 ± 0.54	25.46 ± 1.57	45.34 ± 1.38	42.25 ± 1.26	32.62 ± 2.00	19.07 ± 0.80	14.08 ± 0.94

Table 5: After Fine-tuning: Performance comparison regarding Budgeting percentage on MNIST.

	Epi-Wrapper					
Hybrid INN Accuracy (%)	5% (Weights)	10% (Weights)	20% (Weights)	30% (Weights)	40% (Weights)	50% (Weights)
91.12 ± 0.08	91.08 ± 0.09	91.83 ± 0.04	91.84 ± 0.04	91.85 ± 0.13	91.82 ± 0.09	91.50 ± 0.05

Table 6: Classification accuracies for Hybrid-INN and Epi-Wrapper across CIFAR-10 and CIFAR-100 datasets using different Bayesian backbones (BNNF and BNNR) and network architectures (LeNet-5, ResNet-18, VGG-16).

Dataset	Backbone	# Parameters	BNN Baseline	Hybrid-INN	Epi-Wrapper
CIFAR-10	LeNet-5	166.3K	BNNF	11.01 ± 0.40%	45.54 ± 0.03%
			BNNR	–	–
	ResNet-18	9.82M	BNNF	10.00 ± 0.10%	37.66 ± 1.10%
			BNNR	9.57 ± 0.20%	35.66 ± 1.09%
	VGG-16	30.24M	BNNF	9.98 ± 0.21%	30.12 ± 0.78%
			BNNR	9.89 ± 0.60%	31.04 ± 0.69%
CIFAR-100	ResNet-18	9.8M	BNNF	10.42 ± 0.44%	21.09 ± 0.14%
			BNNR	10.08 ± 0.34%	22.87 ± 0.11%
	VGG-16	30.24M	BNNF	9.01 ± 0.56%	19.98 ± 1.30%
			BNNR	10.01 ± 0.32%	20.10 ± 1.21%

A.6 Fine-tuning on Large-scale models

We modulate the spread of Bayesian neural network weights by applying a layer-specific scaling factor k to the standard deviation when constructing interval bounds. This technique serves to regulate the initial uncertainty of model parameters. A similar concept exists in variational Bayesian inference, where a *tempering parameter* τ is introduced to control the contribution of the likelihood in the posterior distribution [42]. Lower values of τ (or equivalently, of k in our case) lead to sharper, more concentrated posteriors, enhancing training stability and convergence. This calibration mechanism ensures a well-behaved uncertainty estimate, particularly important in interval neural networks (INNs) where the width of intervals directly influences both prediction confidence and robustness.