Uncertainty Quantification using Bayesian Neural Networks

Samir Gupta
Department of Computer Science
Vanderbilt University
Nashville, TN 37203
samir.a.gupta@vanderbilt.edu

Celestine Akpanoko
Department of Computer Science
Vanderbilt University
Nashville, TN 37203
celestine.e.akpanoko@vanderbilt.edu

Abstract

This study discusses the challenge of measuring the level of uncertainty in machine learning predictions, specifically in relation to applications such as predicting household power consumption. This study explores the application of Bayesian Neural Networks, specifically utilizing Monte Carlo Dropout and Deep Ensembles, to improve the precision and dependability of uncertainty estimations. The paper performs experiments on both a simulated dataset and an actual home power usage dataset, utilizing thorough hyperparameter and Bayesian optimization techniques to enhance the models. The results of our study demonstrate that Deep Ensembles exhibit superior performance on different datasets, underscoring their adaptability and reliability in quantifying uncertainty. Nevertheless, Monte Carlo Dropout exhibits a greater susceptibility to the particular structure and optimization of the neural network. Therefore, this study highlights a preference for Deep Ensembles over Monte Carlo Dropout in several machine-learning applications.

1 Introduction

Quantifying uncertainty is a crucial problem in the field of machine learning, especially when models are used in real-world situations where decisions based on predictions can have substantial impacts [1]. Conventional methods frequently offer point estimates without indicating the level of uncertainty, which can be misleading in intricate scenarios [2]. In response to this, Bayesian Neural Networks (BNNs) have emerged as a potent tool, providing a probabilistic viewpoint on modeling [3]. Methods such as Monte Carlo Dropout, as developed by Gal and Ghahramani in 2016 [4], and Deep Ensembles, introduced by Lakshminarayanan et al. in 2017 [5], have made significant progress in this area. Monte Carlo Dropout (MCDO) offers a pragmatic approach to approximate Bayesian inference, allowing models to estimate uncertainty in predictions. On the other hand, Deep Ensembles (DPs) utilize numerous models to enhance the accuracy and dependability of these uncertainty estimates.

The effectiveness of these approaches is especially evident in domains such as household power consumption forecasting, where accuracy in forecasts is vital for efficient energy management and grid stability [6, 7]. Typical models in this field sometimes struggle to account for the inherent unpredictability of forecasts, resulting in possible overconfidence, particularly when incorporating volatile renewable energy sources [8]. The utilization of BNNs, which involve MCDO and DPs, effectively tackles this problem by not only generating predictions but also accurately assessing the confidence levels associated with these predictions. Adopting this method is crucial for dealing with the fluctuations and uncertainties that naturally occur in home power consumption data, including changes in user demand and environmental circumstances, as emphasized in recent research [9].

The objective of this paper is to utilize Monte Carlo Dropout and Deep Ensembles to accurately measure uncertainty. We experiment with a synthetic toy dataset and a household power consumption

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

dataset. We employ a thorough exploration of hyperparameters and utilize Bayesian optimization to refine the models for best performance. We utilize MCDO for emulating Bayesian posterior sampling and leverage DPs to capture a spectrum of potential outcomes and their associated uncertainty. The performance of the models is assessed via the utilization of measures such as mean squared error (MSE), root mean squared error (RMSE), and mean absolute error (MAE). These metrics offer valuable insights into the accuracy and ability to quantify the uncertainty of the models.

The subsequent sections of the paper are structured in the following manner: Section 2 explores the existing research and literature, offering a contextual framework for our approach. Section 3 provides a comprehensive explanation of the techniques used, specifically focusing on the application of MCDO and DPs in our BNNs. Section 4 outlines the experiments performed, which encompassed the exploration of hyperparameters, Bayesian optimization, and the assessment of the models using several metrics. Section 5 provides an analysis of the findings and the conclusions derived from this study, emphasizing the significance and possible future paths of this research in the domain of machine learning and energy prediction.

2 Related Work

2.1 Monte Carlo Dropout

Monte Carlo Dropout (MCDO) is a vital technique used to estimate uncertainty in neural networks. It provides a sophisticated method to improve the resilience and dependability of models. The use of dropout throughout both the training and inference stages is a prevalent approach that has been widely embraced in diverse study fields to enhance comprehension and measurement of uncertainty. The importance of this rests in its ability to offer a practical estimation of model uncertainty, a crucial factor in the creation of resilient and dependable AI systems.

A variety of studies have recently showcased the latest progress in this subject [10, 11, 12, 13, 14, 15]. Zhang et al. [10] showcased the success of MCDO in a physics-informed strategy to deduce Terzaghi's consolidation theory from noisy data, highlighting its utility in complicated contexts. Li et al. [11] introduced a Bayesian Convolutional Recurrent Neural Network (CRNN) in the field of chemical kinetics. This model utilized MCDO to quantify uncertainty and emphasizes variational inference to improve efficiency. Additionally, Basora et al. [12] highlighted the need for optimizing variational and learning parameters in Bayesian neural networks. They conducted a comparison of several inference algorithms, such as MCDO, to enhance the overall performance. These examples highlight the adaptability of MCDO in addressing intricate, data-centric challenges in several scientific domains. Kim et al. [13] investigated the application of MCDO in deep convolutional neural networks for myocardial segmentation in medical imaging. They emphasized its potential in precision-critical tasks. Similarly, Silvestro et al. [14] conducted a comparison of traditional neural networks employing Monte Carlo Dropout (MCDO) and Bayesian Neural Networks (BNNs). The results indicated that BNNs typically exhibited superior performance compared to traditional neural networks, highlighting the importance of meticulous technique selection for uncertainty estimation. The versatility of MCDO in object recognition using a Fully Spiking Hybrid Neural Network demonstrates the method's flexibility [15].

MCDO has a broad range of applicability across several disciplines [16, 17, 18, 19, 20]. In their study, Wang et al. [16] developed a framework for predicting Click-Through Rate. They employed Markov Chain Monte Carlo (MCDO) to estimate posterior parameter distributions, demonstrating its effectiveness in the field of digital marketing. Hu et al. [17] introduced a new approach to forecast the decline in performance of Proton Exchange Membrane Fuel Cells. They included Model Confidence Distributions (MCDO) to measure the level of uncertainty in their predictions. In their study, Ng et al. [18] conducted a thorough analysis that compared Bayesian and non-Bayesian approaches, specifically focusing on MCDO, in the context of segmentation neural networks. The researchers evaluated many performance measures to assess the effectiveness of these techniques. In 2023, Yang and colleagues presented LEDO, a framework that can be applied to any model and is used for detecting and correcting label errors. They showed that this framework, which uses MCDO, is effective across different tasks and datasets [19]. In addition, Wang et al. [20] introduced SeqUST, a self-training framework for Natural Sequence Labeling. This framework utilizes MCDO (Monte Carlo Dropout) in a Bayesian neural network to estimate uncertainty at the token level. These papers

emphasize the increasing use of MCDO, emphasizing its significance in improving the reliability and credibility of AI systems in various decision-making processes.

2.2 Deep Ensemble

The approach of Deep Ensemble in machine learning has become well-known for its efficacy in quantifying uncertainty (UQ). This approach entails employing numerous models or neural networks to reflect the underlying variety in predictions, thereby yielding a more thorough comprehension of uncertainty. The Deep Ensemble model excels at differentiating between aleatoric uncertainties, which are connected to the data, and epistemic uncertainties, which are related to the model itself. This provides a sophisticated understanding of the inherent uncertainty in prediction models. Deep Ensemble utilizes the aggregation of predictions from several models to yield more dependable and resilient estimates compared to techniques that rely on a single model.

Salem et al. [21] showcased the effectiveness of Deep Ensemble in producing prediction intervals in addition to point estimates. Their approach, which combines prediction intervals using a split normal mixture, successfully captures both aleatoric and epistemic uncertainty. This method is essential in situations when comprehending the full spectrum of potential results is as significant to the actual predictions. Egele et al. [22] presented AutoDEUQ, a mechanized method for constructing ensembles of deep neural networks. By using the rule of total variance, they broke down the predictive variation into aleatoric and epistemic components, providing a comprehensive comprehension of the origins of prediction variability.

Roshanzamir et al. [23] investigated the utilization of Deep Ensemble in medical imaging. They employed these techniques to quantify uncertainty in the segmentation of medical images. Their research explored the correlation between these uncertainties and the variety among raters, providing insights into the effects of various neural network structures on the uncertainty of the model. Tang et al. [24] utilized Deep Ensemble methods to forecast the movement of autonomous cars. Their findings highlights the significance of the technique in safety-critical systems, where a precise comprehension of uncertainty limitations is crucial for making well-informed decisions.

Charpentier et al. [25] proposed a framework for separating uncertainty in reinforcement learning by utilizing Deep Ensemble techniques in RL models. Curi et al. [26] supported this method, suggesting the Robust Hallucinated Upper-Confidence RL algorithm. This algorithm, based on models, differentiates between the two forms of uncertainty. In their study, Kook et al. [27] introduced a new transformation ensemble that combines probabilistic predictions and measures both aleatoric and epistemic uncertainty. This ensemble generates minimax optimum predictions in certain circumstances.

These studies emphasize the increasing significance of Deep Ensemble in several fields for accurate measurement of uncertainty. Deep Ensemble provides reliable insights into the variability of predictions. Its usefulness may be further improved by merging it with techniques such as MCDO. MCDO, renowned for its straightforward integration into a unified model framework, can enhance Deep Ensemble by offering a more comprehensive comprehension of uncertainty within individual models. The integration of Deep Ensemble with MCDO presents a comprehensive method for quantifying uncertainty, capitalizing on the advantages of ensemble-based variability analysis and detailed uncertainty estimation inside individual models. An integrated approach would be especially advantageous in intricate, high-stakes situations when it is essential to have a thorough comprehension of both the wide range of potential results and the detailed forecasts made by each separate model.

3 Methods

3.1 Datasets Used

In this section of the paper we describe the datasets used. We divide this section into two parts; first - Synthetic Datasets Generation wherein we describe the datasets and the method used for generating them; second - Real World Dataset here we describe the dataset used to show case the real world use case of our models.

3.1.1 Synthetic Datasets

In this study, we primarily utilized three synthetic datasets. The choice of these synthetic datasets was driven by the need to establish a controlled environment where the variables could be manipulated systematically to understand their impact on the model's performance. In each of these datasets we use a single feature (low dimension) for forecasting and quantifying the uncertainties. Each dataset was meticulously crafted such that the uncertainties quantified by our models can be easily verified and interpreted by humans. The use of these datasets was primarily done to ensure that our models provided the expected results. Figures 1a, 1b and 1c depict the three datasets respecticely. Below are the set of equations used to describe our synthetic datasets:

- Synthetic Dataset 1:
  - Features: $\left\{x \mid x = i \times \frac{2.5}{N},\ i = 0, 1, \ldots, N-1\right\}$, where N is the number of data points.
  - Target: $\left\{\sin(2 \times X_1) + Noise_{Mag} \times (rand(N) - \frac{1}{2})\right\}$, where $Noise_{Mag}$ is noise that will be added to the dataset.
- Synthetic Dataset 2:
  - Features: $\left\{x^5 \mid x = i \times \frac{1}{N},\ i = 0, 1, \ldots, N-1\right\}$, where N is the number of data points.
  - Target: $\left\{\cos(8 \times X_1) + Noise_{Mag} \times (rand(N) - \frac{1}{2})\right\}$, where $Noise_{Mag}$ is noise that will be added to the dataset.
- Synthetic Dataset 3:
  - Features: $\{X_1 \times sign(X_1) \times X_1\}$, where $sign(X_1)$ is a function used to get the sign of the value of X_1.
  - Target: $\left\{\cos(8 \times X_1) + Noise_{Mag} \times (rand(N) - \frac{1}{2})\right\}$, where $Noise_{Mag}$ is noise that will be added to the dataset.

3.1.2 Real-World Dataset (High Dimension)

Following the successful application and validation of our model on these synthetic datasets, we progressed to testing it on a real-world dataset. To this end, we selected a comprehensive dataset available on Kaggle, focused on household power consumption prediction. This is a time-series dataset that provided us with a high dimension set of features (7 features). Using these features we forecast the power consumption rate in a household.

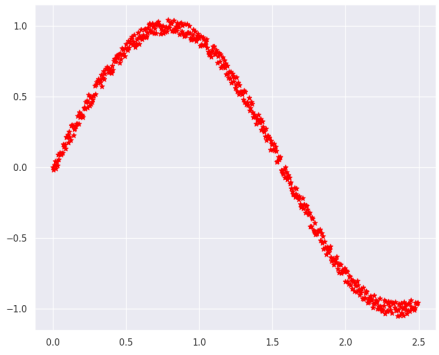
The real-world dataset offered a critical testing ground, allowing us to evaluate the model's effectiveness and accuracy in practical scenarios. Using this dataset we can show how the epistemic uncertainty changes in a time series dataset. We model our epistemic uncertainty graphs with the X-axis as time and y-axis as the target power consumption rate. The plot for this dataset is represented in figure 1d. The features used in this dataset are, *Global active power*, *Global reactive power*, *Voltage Global intensity*, *Sub metering 1*, *Sub metering 2* and *Sub metering 3* all these 7 features were taken for the previous hour to forecast the *Global active power* for the next hour.

3.2 Monte Carlo Drop out

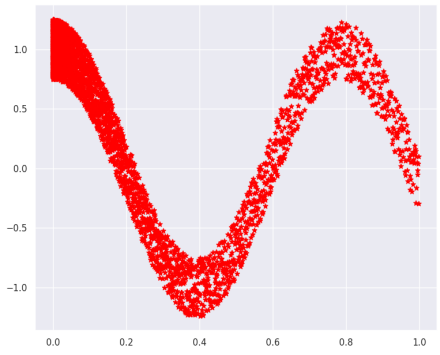
Monte Carlo Dropout, as developed by Gal and Ghahramani [4], is a notable progress in the domain of deep learning, specifically in the realm of BNNs. This methodology utilizes dropout, which is often used as a regularization technique, as a means of approximating Bayesian inference. It offers a computationally efficient way to describe uncertainty.

3.2.1 Bayesian Neural Networks and Variational Inference

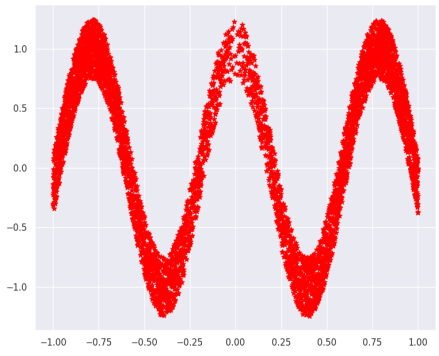
BNNs incorporate uncertainty by placing a prior distribution over their weights, enabling probabilistic inference about model predictions. However, the computational complexity of BNNs often poses a significant challenge. Monte Carlo Dropout emerges as a variant of variational inference, which approximates probability densities through optimization. The objective of variational inference is to find a distribution $q_\theta(w)$ over parameters w that is close to the true posterior $p(w|\mathcal{D})$, typically by minimizing the Kullback-Leibler (KL) divergence.



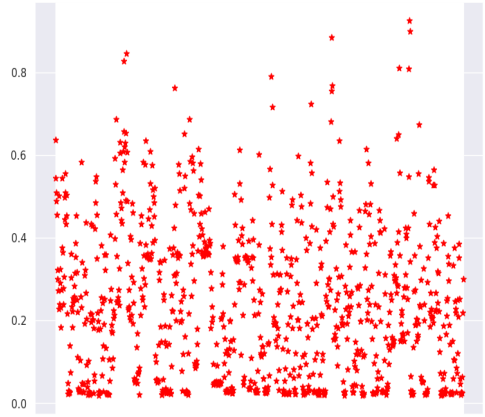
(a) Synthetic Dataset 1



(b) Synthetic Dataset 2



(c) Synthetic Dataset 3



(d) Household Power Consumption

Figure 1: Dataset Visualization Plots

3.2.2 Monte Carlo Dropout as Variational Inference

In the Monte Carlo Dropout framework, the approximating distribution for the weights of each layer i, $q_\theta(W_i)$, is defined as:

$$W_i = M_i \cdot diag([z_{i,j}]_{j=1}^{K_i}) \tag{1}$$

$$z_{i,j} \sim Bernoulli(p_i) \quad for \quad i = 1, \ldots, L, j = 1, \ldots, K_{i-1} \tag{2}$$

where $z_{i,j}$ is a Bernoulli random variable determining the dropout of connected inputs, and p_i represents the probability of retaining an input, inversely related to the traditional dropout rate.

The integral in the ELBO is approximated by minimizing typical loss functions with L2 regularization, expressed as:

$$L_{dropout} = \frac{1}{N} \sum_{i=1}^{N} E(y_i, \hat{y}_i) + \lambda \sum_{i=1}^{L} \left(\frac{\|W_i\|_2^2}{2} + \frac{\|b_i\|_2^2}{2} \right) \tag{3}$$

3.2.3 Predictive Posterior and Model Uncertainty

The predictive probability of the deep Gaussian Process (GP) model, integrated with respect to the finite rank covariance function parameters ω, and given some precision parameter $\tau > 0$, can be parameterized as:

$$p(y|x, X, Y) = \int p(y|x, \omega) p(\omega|X, Y) d\omega \tag{4}$$

The posterior distribution $p(\omega|X,Y)$ is intractable, so we use $q(\omega)$, a distribution over matrices whose columns are randomly set to zero, to approximate the intractable posterior. This variational distribution $q(\omega)$ is highly multimodal, inducing strong joint correlations over the rows of the matrices W_i.

The KL divergence between the approximate posterior $q(\omega)$ and the posterior of the full deep GP, $p(\omega|X,Y)$, is minimized. This is expressed as:

$$-\int q(\omega)\log p(Y|X,\omega)d\omega + KL(q(\omega)||p(\omega)) \tag{5}$$

Each term in the sum is approximated by Monte Carlo integration with a single sample $\omega^n \sim q(\omega)$ to get an unbiased estimate $-\log p(y_n|x_n,\omega^n)$. The model precision τ is used to scale the result, leading to the objective:

$$L_{GP-MC} \propto \frac{1}{N}\sum_{n=1}^{N} -\log p(y_n|x_n,\omega^n)\tau + \sum_{i=1}^{L}\left(\frac{p_i}{l^2 2\tau N}\|M_i\|_2^2 + \frac{l^2}{2\tau N}\|m_i\|_2^2\right) \tag{6}$$

This formulation demonstrates that the dropout goal successfully reduces the Kullback–Leibler divergence between an estimated distribution and the posterior distribution of a deep Gaussian process. The model uncertainty is determined by assessing the variation in the forecasts across several dropout realizations, which serves as an indicator of the level of trust in the model's predictions.

3.3 Deep Ensemble

The Deep Ensemble method, proposed by Lakshminarayanan, Pritzel, and Blundell [5], represents a notable improvement in the assessment of prediction uncertainty in neural networks, specifically for regression applications. This methodology deviates from conventional Bayesian approaches, prioritizing simplicity and scalability, making it very suitable for large-scale applications. The process entails individually training numerous neural networks and combining their outputs, resulting in improved prediction performance and reliable estimation of predictive uncertainty.

In the deep ensemble framework, the training dataset D consists of N i.i.d. data points $D = \{x_n, y_n\}_{n=1}^N$, where $x \in \mathbb{R}^D$ represents D-dimensional features. For classification tasks, y is one of K classes, $y \in \{1,\ldots,K\}$, and for regression tasks, y is real-valued, $y \in \mathbb{R}$. The neural network models the probabilistic predictive distribution $p_\theta(y|x)$ over the labels, with θ as the network parameters. The training process involves using a proper scoring rule as the training criterion, implementing adversarial training to smooth predictive distributions, and training an ensemble of M neural networks, denoted by $\{\theta_m\}_{m=1}^M$.

Scoring rules play a crucial role in measuring the quality of predictive uncertainty. A scoring rule assigns a numerical score to a predictive distribution $p_\theta(y|x)$, rewarding better-calibrated predictions over worse. The scoring rule is a function $S(p_\theta,(y,x))$ that evaluates the quality of the predictive distribution $p_\theta(y|x)$ relative to an event $y|x \sim q(y|x)$, where $q(y,x)$ denotes the true distribution. The expected scoring rule is then $S(p_\theta,q) = \int q(y,x)S(p_\theta,(y,x))dydx$. A proper scoring rule is one where $S(p_\theta,q) \leq S(q,q)$, with equality if and only if $p_\theta(y|x) = q(y|x)$. Neural networks are trained to encourage calibration of predictive uncertainty by minimizing the loss $L(\theta) = S(p_\theta,q)$.

In the context of regression, neural networks typically output a single value, $\mu(x)$, optimized to minimize the mean squared error (MSE) on the training set. However, MSE does not capture predictive uncertainty. To address this, deep ensembles use a network architecture that outputs two values in the final layer: the predicted mean $\mu(x)$ and variance $\sigma^2(x) > 0$. This dual-output approach treats the observed value as a sample from a heteroscedastic Gaussian distribution with the predicted mean and variance. The training objective becomes minimizing the negative log-likelihood criterion:

$$-\log p_\theta(y_n|x_n) = \log\sigma_\theta^2(x) + \frac{(y-\mu_\theta(x))^2}{2\sigma_\theta^2(x)} + constant. \tag{7}$$

The quantification of predictive uncertainty in deep ensembles for regression is achieved by measuring the variance among the predictions made by the individual members of the ensemble. The predictive distribution of the ensemble is obtained by averaging the predictions of the different models for a given input x, where the variance of these forecasts represents the uncertainty. This technique offers

a reliable way to estimate prediction uncertainty in regression tasks in deep learning, without using Bayesian methods. It provides a thorough understanding of the model's performance and presents a convincing alternative to more intricate Bayesian approaches

4 Experiments

In this section of the paper we discuss the various experiments that we ran over our datasets. The primary goal of the paper is to quantify the epistemic uncertainty present in the models. However, the main step is to reduce the epistemic uncertainty present, this can be achieved by tuning the parameters to best suit the model. In the following subsection we discuss the hyperparameter search over our datasets.

4.1 Hyperparameter Search

We performed hyperparameter search on total of 8 models (2 models for each of our 4 datasets) and the best hyperparameters for our MCDO and Deep ensemble models were determined by tuning the parameters listed in Table 1.

Hyperparameter Name	**Range**
Number of Neurons	[1,200]
Number of layers	[2,10]
Dropout Rate	[0.1,0.5]
Learning Rate	[0.0001, 0.1]
Patience	[1, 50]
Batch Size	[1,20]

Table 1: Hyperparmeters Tuned for MCDO and Deep Ensemble models

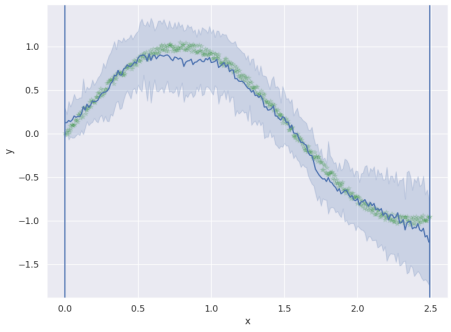
In table 1 *hidden size* represents the number of neurons for each dense layer after the input layers. We ran the hyperparameter search for a total of 500 iterations with the help of Ray tuner [28]. Each model was trained for a total of 50 epochs, with early stopping over validation loss and a patience which was determined using hyperparamter search. The best hyperparameters were taken by taking the minimum validation loss for each model. Table 2 highlights the best hyperparameters acquired for each dataset and model.

4.1.1 Bayesian Optimziation for hyperparamter search

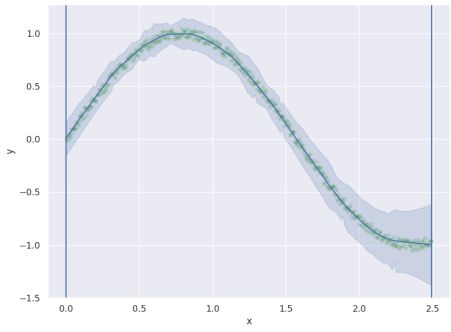
In this paper, we use Bayesian optimization for hyperparameter search of our 8 machine learning models, leveraging its ability to intelligently navigate the hyperparameter space. In contrast to traditional grid or random search methods, Bayesian optimization uses a probabilistic model to predict the performance of different hyperparameter combinations and iteratively refines this model based on evaluation results. This approach not only enhances the efficiency of the search process, significantly reducing the computational resources and time required, but also often leads to superior model performance by effectively balancing the exploration and exploitation trade-off in the hyperparameter space.

Model and Data	**batch size**	**drop out**	**num layers**	**hidden size**	**lr**	**patience**	**epochs**
MCDO - Real Data	1	0.1645	10	29	0.00025	27	500
DE - Real Data	1	0	2	80	0.00088	18	500
MCDO - Data 1	1	0.1280	4	26	0.01014	22	500
DE - Data 1	1	0	3	33	0.0001	29	500
MCDO - Data 2	1	0.2701	3	53	0.00487	26	500
DE - Data 2	1	0	9	99	0.0001	14	500
MCDO - Data 3	1	0.1	10	131	0.0001	17	500
DE - Data 3	1	0	5	68	0.00187	20	500

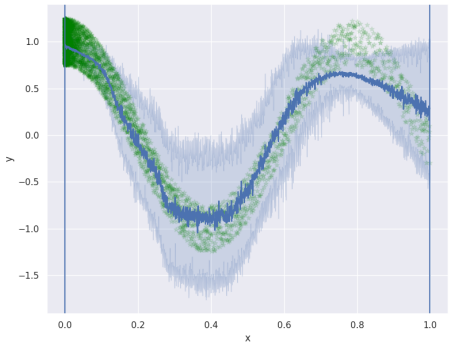
Table 2: Hyperparameter for various models and datasets.



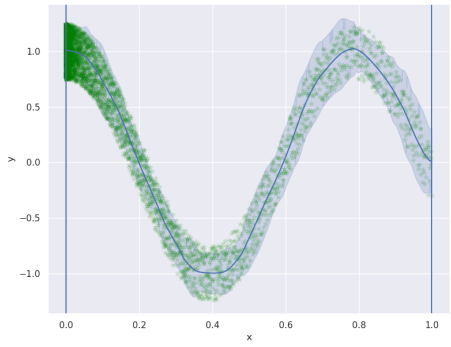
(a) Syn Dataset 1 - MCDO



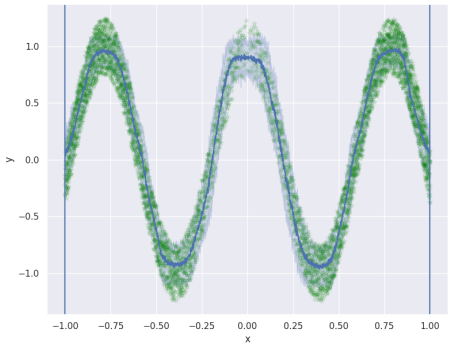
(b) Syn Dataset 1 - Deep Ensemble



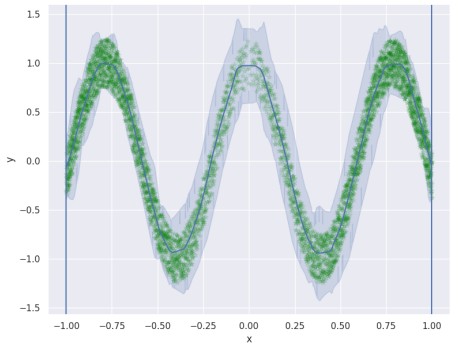
(c) Syn Data 2 - MCDO



(d) Syn Data 2 - Deep Ensemble

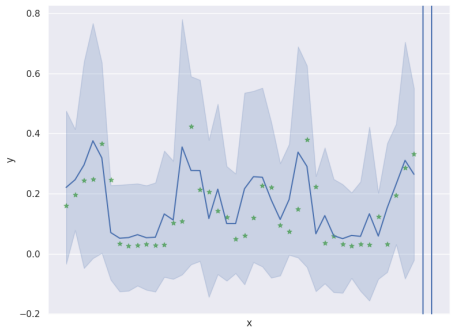


(e) Syn Dataset 3 - MCDO



(f) Syn Dataset 3 - Deep Ensemble

(g) Household Power Consumption - MCDO



(h) Household Power Consumption - Deep Ensemble

Figure 2: Epistemic Uncertainty Quantification Plots

4.2 Epistemic Uncertainty Quantification

Following the completion of our hyperparameter search, we proceeded to train our models using the best hyperparameters fpr each of our 8 models as mentioned in table 2. We use Monte Carlo Dropout (MCDO) to quantify the epistemic uncertainty. MCDO is a variant of the traditional dropout technique, which involves activating dropout during both the training and the inferential (testing) phases, allowing us to approximate Bayesian posterior distributions. This method provides a dynamic approach to evaluate the model's confidence (quantify the uncertainty in the model) in its predictions, by enabling the estimation of epistemic uncertainty.

Further, we explored the efficacy of Deep Ensemble models, a robust technique that creates a composite model from multiple individual models. We trained a total of 5 models as the number of ensembles. This method uses these 5 independently trained models to quantify the epistemic uncertainty by aggregating the disparate predictions, we could assess the variance and thereby the confidence intervals around the model's outputs.

Graphical representations of uncertainty were plotted for both MCDO and Deep Ensemble models as depicted in figure 2. In these plots the *green* points represent the test data points. The *dark-blue* line highlights the mean of the predictions made using MCDO and Deep ensemble models, finally the *light-blue* shaded region indicates 2 standard deviations $\pm$ mean and shows the epistemic uncertainty in the model's predictions. For the synthetic datasets (figures 2a, 2b, 2c, 2d, 2e and 2f), the X-axis represents the single feature that was used to train the models and Y-axis indicates the target prediction values. Similarly, for our real world dataset (figures 2g and 2h) i.e. the household power consumption dataset the Y-axis is used to depict the target i.e. *Global active power*, whereas, the X-axis highlights the time for the corresponding target values, we only show randomly picked 40 consecutive values from our test dataset. This was done for clarity and interpretability of the graphs, the plot size was deliberately truncated to present a more comprehensible visual representation. This decision was driven by the sheer volume of data points, which, when plotted in full, resulted in an overly complex and dense graphical output.

$$MSE = \frac{1}{N}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{8}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{9}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{10}$$

We also calculate the mean squared error, root mean squared error, and mean absolute error using the formulae 8, 9 and 10. The results for all the 8 models are mentioned in table 3.

Data and Model	MSE - Train	RMSE - Train	MAE - Train	MSE - Val	RMSE - Val	MAE - Val	MSE - Test	RMSE - Test	MAE - Test
MCDO - Real Data	0.010	0.098	0.065	0.008	0.090	0.061	0.007	0.085	0.058
DE - Real Data	0.009	0.096	0.068	0.008	0.090	0.067	0.008	0.087	0.065
MCDO - Data 1	0.006	0.080	0.064	0.017	0.130	0.085	0.007	0.081	0.062
DE - Data 1	0.001	0.030	0.026	0.002	0.047	0.032	0.001	0.030	0.026
MCDO - Data 2	0.030	0.173	0.143	0.048	0.219	0.176	0.047	0.217	0.175
DE - Data 2	0.022	0.150	0.128	0.025	0.157	0.134	0.024	0.155	0.132
MCDO - Data 3	0.023	0.152	0.128	0.025	0.158	0.134	0.024	0.156	0.132
DE - Data 3	0.022	0.147	0.125	0.024	0.154	0.131	0.022	0.150	0.128

Table 3: Performance metrics for various models and datasets.

5 Discussion and Conclusion

From figure 2, it can be inferred that deep ensemble models are considerably better or perform as good as MCDO at quantifying uncertainty for all the four datasets described in our work. For Synthetic Dataset 1, we added a few extra data points at the bottom right corner of the wave to increase the epistemic uncertainty. From figure 2b it we can see that the uncertainty quantified in that region

is a lot higher as compared to the remainder of the data points in the visualization plot. Similarly, Deep ensemble perform much better in quantifying uncertainty for synthetic datasets 2 and 3 as well, wherein, the areas with fewer data points have larger uncertainty widths as shown in figures 2d, 2c, 2f and 2c. Deep ensemble models quantify the model's uncertainty in the real dataset (household power consumption) far better as compared to MCDO as well, as the distribution of the datapoints changes considerably throughout the day, deep ensemble models are able to capture this change in the predictions and show how the predictions can be affected depending on the time of the day. From figures 2h and 2g deep ensemble model does not overfit as compared to MCDO.

Deep ensemble are generally considered superior to Monte Carlo dropout for quantifying uncertainty in machine learning models for several reasons. **(a)** Deep ensembles consist of multiple independently trained models (in our case we train 5 different models), this offers a broader perspective on the data. This diversity results in more reliable and varied predictions compared to the single-model approach of Monte Carlo dropout. **(b)** Deep ensembles aggregate predictions from multiple models which can effectively reduce the bias that might be inherent in a single model (Monte carlo drop out). **(c)** Deep ensembles are more robust to overfitting compared to Monte Carlo Drop out that makes use of drop out to reduce overfitting. This is because the overfitting of one model in the ensemble may be balanced out by others, leading to a more generalized performance on unseen data, this is captured by figures 2h and 2g.

Through our work we show that Deep ensembles are more effective across a wide range of datasets, demonstrating their versatility and robustness in quantifying uncertainty. Monte Carlo dropout, while useful, can be more sensitive to the specific architecture and tuning of the neural network. In conclusion, Deep ensembles are favored over Monte Carlo dropout in many machine learning applications because they offer a more reliable, varied, and calibrated method of assessing uncertainty.

References

[1] Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, Muhammad Shahzad, Wen Yang, Richard Bamler, and Xiao Xiang Zhu. A survey of uncertainty in deep neural networks.

[2] Eyke Hüllermeier and Willem Waegeman. Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods. 110(3):457–506.

[3] H. M. Dipu Kabir, Abbas Khosravi, Mohammad Anwar Hosen, and Saeid Nahavandi. Neural network-based uncertainty quantification: A survey of methodologies and applications. *IEEE Access*, 6:36218–36234, 2018.

[4] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

[5] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

[6] Mohammed F Alsharekh, Shabana Habib, Deshinta Arrova Dewi, Waleed Albattah, Muhammad Islam, and Saleh Albahli. Improving the efficiency of multistep short-term electricity load forecasting via r-cnn with ml-lstm. *Sensors*, 22(18):6913, 2022.

[7] Alper Ozcan, Cagatay Catal, and Ahmet Kasif. Energy load forecasting using a dual-stage attention-based recurrent neural network. *Sensors*, 21(21):7115, 2021.

[8] Fotios Petropoulos, Daniele Apiletti, Vassilios Assimakopoulos, Mohamed Zied Babai, Devon K Barrow, Souhaib Ben Taieb, Christoph Bergmeir, Ricardo J Bessa, Jakub Bijak, John E Boylan, et al. Forecasting: theory and practice. *International Journal of Forecasting*, 38(3):705–871, 2022.

[9] Mingyang Sun, Tingqi Zhang, Yi Wang, Goran Strbac, and Chongqing Kang. Using bayesian deep learning to capture uncertainty for residential net load forecasting. *IEEE Transactions on Power Systems*, 35(1):188–201, 2019.

[10] Pin Zhang, Zhen-Yu Yin, and Brian Sheil. A physics-informed data-driven approach for consolidation analysis. *Géotechnique*, pages 1–12, 2023.

[11] Qiaofeng Li, Huaibo Chen, Benjamin C Koenig, and Sili Deng. Bayesian chemical reaction neural network for autonomous kinetic uncertainty quantification. *Physical Chemistry Chemical Physics*, 25(5):3707–3717, 2023.

[12] Luis Basora, Arthur Viens, Manuel Arias Chao, and Xavier Olive. A benchmark on uncertainty quantification for deep learning prognostics. *arXiv preprint arXiv:2302.04730*, 2023.

[13] Yoon-Chul Kim, Khu Rai Kim, and Yeon Hyeon Choe. Automatic myocardial segmentation in dynamic contrast enhanced perfusion mri using monte carlo dropout in an encoder-decoder convolutional neural network. *Computer methods and programs in biomedicine*, 185:105150, 2020.

[14] Daniele Silvestro and Tobias Andermann. Prior choice affects ability of bayesian neural networks to identify unknowns. *arXiv preprint arXiv:2005.04987*, 2020.

[15] Biswadeep Chakraborty, Xueyuan She, and Saibal Mukhopadhyay. A fully spiking hybrid neural network for energy-efficient object detection. *IEEE Transactions on Image Processing*, 30:9014–9029, 2021.

[16] Xiaowei Wang and Hongbin Dong. Click-through rate prediction and uncertainty quantification based on bayesian deep learning. *Entropy*, 25(3):406, 2023.

[17] Yanyan Hu, Li Zhang, Yunpeng Jiang, Kaixiang Peng, and Zengwang Jin. A hybrid method for performance degradation probability prediction of proton exchange membrane fuel cell. *Membranes*, 13(4):426, 2023.

[18] Matthew Ng, Fumin Guo, Labonny Biswas, Steffen E Petersen, Stefan K Piechnik, Stefan Neubauer, and Graham Wright. Estimating uncertainty in neural networks for cardiac mri segmentation: A benchmark study. *IEEE Transactions on Biomedical Engineering*, 2022.

[19] Xiao Yang, Ahmed K Mohamed, Shashank Jain, Stanislav Peshterliev, Debojeet Chatterjee, Hanwen Zha, Nikita Bhalla, Gagan Aneja, and Pranab Mohanty. Improving opinion-based question answering systems through label error detection and overwrite. *arXiv preprint arXiv:2306.07499*, 2023.

[20] Jianing Wang, Chengyu Wang, Jun Huang, Ming Gao, and Aoying Zhou. Uncertainty-aware self-training for low-resource neural sequence labeling. *arXiv preprint arXiv:2302.08659*, 2023.

[21] Tárik S Salem, Helge Langseth, and Heri Ramampiaro. Prediction intervals: Split normal mixture from quality-driven deep ensembles. In *Conference on Uncertainty in Artificial Intelligence*, pages 1179–1187. PMLR, 2020.

[22] Romain Egele, Romit Maulik, Krishnan Raghavan, Bethany Lusch, Isabelle Guyon, and Prasanna Balaprakash. Autodeuq: Automated deep ensemble with uncertainty quantification. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 1908–1914. IEEE, 2022.

[23] Parinaz Roshanzamir, Hassan Rivaz, Joshua Ahn, Hamza Mirza, Neda Naghdi, Meagan Anstruther, Michele C Battié, Maryse Fortin, and Yiming Xiao. How inter-rater variability relates to aleatoric and epistemic uncertainty: a case study with deep learning-based paraspinal muscle segmentation. In *International Workshop on Uncertainty for Safe Utilization of Machine Learning in Medical Imaging*, pages 74–83. Springer, 2023.

[24] Xiaolin Tang, Kai Yang, Hong Wang, Jiahang Wu, Yechen Qin, Wenhao Yu, and Dongpu Cao. Prediction-uncertainty-aware decision-making for autonomous vehicles. *IEEE Transactions on Intelligent Vehicles*, 7(4):849–862, 2022.

[25] Bertrand Charpentier, Ransalu Senanayake, Mykel Kochenderfer, and Stephan Günnemann. Disentangling epistemic and aleatoric uncertainty in reinforcement learning. *arXiv preprint arXiv:2206.01558*, 2022.

[26] Sebastian Curi, Ilija Bogunovic, and Andreas Krause. Combining pessimism with optimism for robust and efficient model-based deep reinforcement learning. In *International Conference on Machine Learning*, pages 2254–2264. PMLR, 2021.

[27] Lucas Kook, Andrea Götschi, Philipp FM Baumann, Torsten Hothorn, and Beate Sick. Deep interpretable ensembles. *arXiv preprint arXiv:2205.12729*, 2022.

[28] Richard Liaw, Eric Liang, Robert Nishihara, Philipp Moritz, Joseph E. Gonzalez, and Ion Stoica. Tune: A research platform for distributed model selection and training.