Characterizing Monte Carlo Dropout to Quantify Uncertainty in Trained Neural Networks

Rafael de S. Luna[1], **Rodrigo de S. Luna**[2], **Daniel R. Figueiredo**[2]

[1] Engineering Department –
Universidade Pitágoras Unopar Anhanguera (Unopar)
Rio de Janeiro, RJ, Brazil

[2]Computer Sciences and Systems Engineering Department
Universidade Federal do Rio de Janeiro (UFRJ)
Rio de Janeiro, RJ, Brazil

{rafaelluna,rluna,daniel}@cos.ufrj.br

Abstract. *Uncertainty quantification is crucial in deep learning applications where reliable decisions depend not only on accuracy but also on predictive confidence. This work investigates Monte Carlo Dropout (MC Dropout), an efficient method to estimate epistemic uncertainty in neural networks. In particular, the influence of its uncertainty parameter is characterized using the California Housing dataset in a regression task. Empirical results reveal that the estimated uncertainty is highly sensitive to this parameter, affecting both the sample average and standard deviation. Interestingly, the error (RMSE) of the model is not monotonic with the uncertainty parameter, indicating an optimal uncertainty value. These findings underscore the need for careful dropout selection during inference and motivate future research on model calibration.*

1. Introduction

Deep learning models have achieved outstanding performance in various regression and classification tasks. However, a major limitation of these models is the lack of reliable uncertainty estimates associated with their predictions. In critical applications such as medical diagnosis, weather forecasting, or autonomous systems, confidence in the model's output can be as important as the prediction itself. In such scenarios, uncertainty quantification becomes a key tool for robust decision-making [Abdar et al. 2021, Kendall and Gal 2017].

The Bayesian approach offers a solid theoretical framework for representing *epistemic uncertainty* that is, uncertainty about the model parameters themselves. This type of uncertainty is linked to the limitations in knowledge and can be reduced with more data or better models [Walker et al. 2003, Der Kiureghian and Ditlevsen 2009]. However, traditional Bayesian inference in deep neural networks is computationally expensive and often intractable at scale [Blundell et al. 2015]. To address this issue, several approximate methods have been proposed, with *Monte Carlo (MC) Dropout* standing out as a practical and efficient approach.

The original formulation of MC Dropout, [Gal and Ghahramani 2016] interpreted dropout as a form of variational inference, where the approximate posterior $q(\theta)$ is defined by applying random Bernoulli masks to neurons with a fixed probability p during training.

This dropout rate implicitly characterizes the structure of $q(\theta)$, which is then sampled during inference to estimate uncertainty. Although the theoretical derivation assumes a fixed dropout rate, used to minimize the Kullback-Leibler divergence between $q(\theta)$ and the true posterior $p(\theta \mid \mathcal{D})$, the authors do not explicitly specify whether the same dropout rate must be maintained at test time. Changing p only during inference alters the sampling distribution over weights, potentially increasing or decreasing the predictive variance. To the best of our knowledge, the impact of this modification on uncertainty quantification, particularly on confidence interval calibration, has not been systematically investigated. This work addresses this gap by empirically evaluating how the uncertainty parameter (dropout rate) at inference influences the epistemic uncertainty estimates provided by MC Dropout.

This study experimentally investigates the influence of the uncertainty parameter (dropout rate) exclusively at inference time, while keeping the trained model fixed. Extensive experiments using the California Housing dataset in a regression task are used to characterize the relationship between the uncertainty parameter and sample average, standard deviation, confidence interval, and estimation error (RMSE). Results indicate that this parameter plays a fundamental role at inference time. In particular, the estimation error (RMSE) is not monotonic with the parameter, suggesting an optimal value for the error. Moreover, increasing the parameter generates larger uncertainties and larger confidence intervals while also degrading the point estimation (sample average). These findings underscore the need for careful dropout selection during inference and motivate future research on model calibration.

2. Background and Related Work

Originally introduced as a regularization technique, the dropout algorithm aims to prevent overfitting in deep neural networks [Srivastava et al. 2014]. During training, each unit is retained with a certain probability p, according to a Bernoulli distribution, which encourages the network to learn redundant and robust representations. The authors observed that intermediate values of p, typically between 0.5 and 0.8, yield the best results, with $p = 0.5$ being a common choice for fully connected layers [Baldi and Sadowski 2013]. This choice reflects a balance between maintaining the network's expressive capacity and reducing overfitting.

Formally, the dropout operation can be modeled by multiplying the activation of each unit by a binary random variable drawn from a Bernoulli distribution:

$$\begin{aligned} z_i &\sim \text{Bernoulli}(1-p), \\ \tilde{h}_i &= z_i \cdot h_i, \end{aligned} \tag{1}$$

where h_i is the activation of the i-th unit in the layer, and $\tilde{h}_i$ is the resulting output after applying dropout. The parameter $p \in [0, 1)$ denotes the dropout rate, i.e., the probability of *dropping* (i.e., setting to zero) a given unit. Consequently, each $z_i \in \{0, 1\}$ is a stochastic binary variable such that:

- $z_i = 1$ with probability $1 - p$: the unit is retained;
- $z_i = 0$ with probability p: the unit is dropped.

This stochastic masking mechanism prevents co-adaptation of feature detectors and encourages robustness in the learned representations. The retained activations $\tilde{h}_i$ are then passed to the subsequent layers.

However, when dropout is employed during inference for the purpose of uncertainty quantification, the role of the parameter p changes significantly. In this context, the objective is no longer regularization, but rather the generation of a distribution of predictions by keeping dropout active across multiple forward passes. This enables the estimation of statistics such as the predictive mean and variance, effectively providing a practical approximation to Bayesian inference.

2.1. The Importance of Uncertainty Quantification

The ability to quantify uncertainty is fundamental in applications where safety, reliability, or explainability are critical requirements. In AI-assisted medical diagnosis, for instance, a high uncertainty estimate can alert the specialist that a given prediction requires closer attention or manual validation [Esteva et al. 2017]. In predictive maintenance, uncertainty can be used to define safe operational windows, while in physical simulations, such as climate or engineering models, its incorporation enables the representation of error margins and confidence intervals associated with complex phenomena.

Moreover, uncertainty quantification is essential for detecting out-of-distribution (OOD) samples, calibrating probabilistic classifiers, implementing active learning strategies (where the model selects the most uncertain samples for annotation), and supporting rejection mechanisms for low-confidence predictions. These aspects make uncertainty estimation an indispensable component of machine learning systems applied to high-stakes domains.

This work aims to systematically assess how different values of p, used exclusively during inference, affect the statistical properties of predictions obtained through MC Dropout. The focus is on understanding how adjusting the dropout rate can modulate the estimated uncertainty without the need to retrain the model.

2.2. Dropout as a Bayesian Sampler

The use of dropout at inference time has been shown to approximate Bayesian inference in deep neural networks [Gal and Ghahramani 2016]. Specifically, this approach corresponds to minimizing the Kullback–Leibler divergence between the true posterior distribution $p(\mathbf{w} \mid \mathcal{D})$ and a variational approximation $q(\mathbf{w})$, where $q(\mathbf{w})$ is implicitly defined by stochastic dropout masks applied during training and inference.

At test time, uncertainty is captured by performing T stochastic forward passes with independently sampled dropout masks, yielding an empirical distribution of predictions:

$$\{\hat{y}^{(t)}\}_{t=1}^{T}, \quad \hat{y}^{(t)} = f_{\mathbf{w}^{(t)}}(\mathbf{x}), \quad \mathbf{w}^{(t)} \sim q(\mathbf{w}), \tag{2}$$

where each $\hat{y}^{(t)}$ is the output of the network under a different dropout configuration. This Monte Carlo sampling procedure enables the estimation of predictive moments such as mean and variance, providing a tractable approximation to Bayesian marginalization. Despite its simplicity and compatibility with standard training pipelines, Monte

Carlo dropout introduces a non-negligible computational overhead at inference time, as multiple forward passes are required per input. Moreover, the variational family induced by dropout is limited in expressiveness, which may hinder the approximation quality in complex posterior landscapes. Recent works have explored alternatives that improve calibration or efficiency, such as deep ensembles [Lakshminarayanan et al. 2017], stochastic weight averaging [Izmailov et al. 2018], and variational approximations using normalizing flows [Louizos et al. 2017], yet MC dropout remains a practical baseline for uncertainty quantification in deep learning.

Uncertainty Estimates. With the T predictions obtained, the predictive mean and variance can be estimated as:

$$\mathbb{E}[y] \approx \frac{1}{T}\sum_{t=1}^{T}\hat{y}^{(t)} \qquad \mathrm{Var}[y] \approx \frac{1}{T}\sum_{t=1}^{T}(\hat{y}^{(t)} - \mathbb{E}[y])^2 \tag{3}$$

The estimated variance can be interpreted as a measure of the model's epistemic uncertainty, as it reflects the sensitivity of predictions to different network samplings. This approach, has low additional computational cost, is compatible with pretrained architectures, and can be applied to both regression and classification tasks.

Unlike in training, the value of p during inference acts as a direct control of the estimated epistemic variance. Higher dropout rates tend to produce greater variability among prediction samples, resulting in wider uncertainty estimates. However, such variation may not faithfully reflect the uncertainty learned by the model, especially if the value of p differs from that used during training. Therefore, although p is traditionally kept fixed, investigating its specific impact during inference is a relevant and underexplored issue in the literature.

2.3. Uncertainty in Machine Learning Models

In machine learning, uncertainty is associated with the degree of confidence a model has when making a prediction. It represents the explicit modeling of the model's lack of knowledge about a given input or its own parameters, which is essential for reliable decision-making. The literature distinguishes two main types of uncertainty: *epistemic* and *aleatoric* [Kendall and Gal 2017, Der Kiureghian and Ditlevsen 2009].

Epistemic uncertainty, also known as model uncertainty, refers to the uncertainty over the model's parameters due to limited observed data. In other words, it arises from a lack of knowledge and can be reduced as more training data becomes available. This type of uncertainty is particularly relevant in domains with scarce labeled examples, such as in medical tasks, where data collection is expensive and sensitive [Walker et al. 2003]. From a Bayesian perspective, this uncertainty can be modeled by treating the network weights $\mathbf{w}$ as random variables with a posterior distribution $p(\mathbf{w} \mid \mathcal{D})$, given the dataset $\mathcal{D}$. The marginal predictive distribution considering this uncertainty is given by:

$$p(y \mid \mathbf{x}, \mathcal{D}) = \int p(y \mid \mathbf{x}, \mathbf{w})\, p(\mathbf{w} \mid \mathcal{D})\, d\mathbf{w}, \tag{4}$$

where $p(y \mid \mathbf{x}, \mathbf{w})$ represents the predictive distribution of the network for fixed weights $\mathbf{w}$, and $p(\mathbf{w} \mid \mathcal{D})$ is the posterior distribution over the weights.

On the other hand, **aleatoric uncertainty**, or irreducible randomness, arises from the inherent variability in the data-generating process. It is caused by natural stochastic factors, such as sensor noise, environmental conditions, or random fluctuations in physical systems, and cannot be reduced even with large amounts of data [Der Kiureghian and Ditlevsen 2009]. This type of uncertainty is associated with the predictive component $p(y \mid \mathbf{x}, \mathbf{w})$, assuming the network weights are fixed. Explicit modeling of this component may involve using parametric distributions in the network outputs. For regression tasks, for example, one can assume that the target y follows a Gaussian distribution:

$$y \sim \mathcal{N}(\mu(\mathbf{x}), \sigma^2(\mathbf{x})), \tag{5}$$

where the network is trained to predict both the mean $\mu(\mathbf{x})$ and variance $\sigma^2(\mathbf{x})$ as functions of the input $\mathbf{x}$. The variance σ^2 directly represents aleatoric uncertainty and can be optimized through the maximization of the log-likelihood of the data [Kendall and Gal 2017].

Understanding and quantifying these two types of uncertainty is fundamental to ensuring robust predictions in sensitive contexts such as medical, financial, and industrial applications. Furthermore, their conceptual separation enables the development of specific approaches for their modeling and mitigation, such as the use of Bayesian techniques for epistemic uncertainty and explicit probabilistic modeling for aleatoric uncertainty.

3. Methodology

In this section, we describe the dataset, the neural network architecture used, the training procedures, and the configuration of the experiments with different dropout rates during the inference phase.

3.1. Uncertainty over θ and Evaluation

Neural networks learn a function approximator $\hat{y} = F(x; \theta)$, where θ represents the learned weights. In critical decision-making domains such as health, climate, finance, and engineering, estimating only the point prediction $\hat{y}$ is not sufficient. It is also essential to quantify the predictive uncertainty, or how reliable the output $F(x; \theta)$ is for a given input x.

From a Bayesian perspective, this requires treating the model parameters θ as random variables with a posterior distribution $p(\theta \mid \mathcal{D})$ given the training data $\mathcal{D}$. The predictive distribution is then defined as the expectation over all possible model configurations, expressed by the integral:

$$\mathbb{E}[y \mid x, \mathcal{D}] = \int F(x; \theta)\, p(\theta \mid \mathcal{D})\, d\theta \tag{6}$$

and the associated predictive variance is given by:

$$\text{Var}[y \mid x, \mathcal{D}] = \int (F(x; \theta) - \mathbb{E}[y \mid x, \mathcal{D}])^2\, p(\theta \mid \mathcal{D})\, d\theta. \tag{7}$$

Since computing these integrals is intractable for deep neural networks, we employ an efficient approximation known as Monte Carlo Dropout (MC Dropout). This technique enables epistemic uncertainty estimation without modifying the network's architecture. By keeping dropout layers active during inference, we generate multiple stochastic forward passes using different dropout masks $\mathcal{M}_i \sim \text{Bernoulli}(p)$, producing samples $\hat{y}^{(i)} = f(x; \mathcal{M}_i)$ for $i = 1, \ldots, N$ where N is the number of independent dropout masks. The predictive mean and variance are then approximated by:

$$\mathbb{E}_p[\hat{y}] \approx \frac{1}{N} \sum_{i=1}^{N} f(x; \mathcal{M}_i) \qquad \text{Var}_p[\hat{y}] \approx \frac{1}{N} \sum_{i=1}^{N} (f(x; \mathcal{M}_i) - \mathbb{E}[\hat{y}])^2 \tag{8}$$

Note that the above estimates for $\mathbb{E}_p[\hat{y}]$ and $\text{Var}_p[\hat{y}]$ depend on the dropout parameter p. Recall that $1 - p$ is the expected ratio of neural units that remain active after the dropout. Intuitively, larger values of p will generate estimates that are more noisy estimates since less units will remain active during sampling.

In what follows, we evaluate how different values of the dropout rate p affect the estimation of epistemic uncertainty under the MC Dropout framework, while keeping the trained model fixed. Although p is commonly treated as a hyperparameter fixed during training and inference, this practice is not mandatory. By modifying p only at inference time, we implicitly change the approximate posterior $q(\theta)$, which may directly influence the predictive average and variance. We therefore conduct an empirical study by systematically varying the inference-time dropout rate and observing its impact on the predictive average and variance, the width of confidence intervals, and the overall calibration quality of the model's outputs.

3.2. Experimental Setup

Data setup. We used the California Housing dataset provided by the the `scikit-learn` library, a common benchmark for regression tasks, which contains demographic and geographic attributes from different regions of California to predict the median house value. The data was split following the standard train/test partition, with 17,000 samples for training (85%) and 3,000 for testing (15%). All input features of the dataset were normalized using standardization (zero mean and unit variance).

Model setup. The model considered is a deep fully-connected neural network composed of five hidden layers with the following number of units: [1000, 2000, 2000, 2000, 1000]. The activation function used was *ReLU*, and weight initialization followed the *LeCun normal* scheme. The architecture includes parameterizable dropout, allowing explicit control of the rate as an additional input to the model. This design enables inference with dropout rates different from those used during training.

The model was trained using the *Mean Squared Error* (MSE) loss function and optimized with the *Adam* optimizer. The dropout rate during training was fixed at $p = 0.5$ for all layers with stochastic regularization. Training was conducted with early stopping, using a patience of 10 epochs based on validation loss (20% of the training set), with a maximum of 200 epochs.

Inference with varying Dropout. After training, the model was evaluated under different dropout rates *applied only during inference*, ranging from $p = 0.1$ to $p = 0.9$

in increments of 0.1. For each rate p, we performed stochastic inference with *Monte Carlo Dropout*, generating T samples per test example with dropout enabled. Using the T independent samples, the predictive mean and standard deviation were then computed for each instance in the test set. The 95% confidence intervals for each instance in the test set was also computed assuming a normal distribution for the prediction, following common practices in previous works [Gal and Ghahramani 2016, Kendall and Gal 2017].

Additionally, specific test instances were selected for qualitative analysis of confidence intervals, constructed with 95% coverage assuming a normal distribution ($\mu \pm 1.96 \cdot \sigma$). These instances correspond to percentiles of the empirical distribution of the ground truth (20-, 40-, 60-, and 80-percentile) capturing different regimes (magnitudes) of the distribution.

4. Results and Discussion

In this section, we analyze the results obtained according the experimental setup described above. Different characteristics of the predictions were analyzed and are presented below.

4.1. Confidence Intervals and Coverage of Ground Truth

The first analysis considers the size the confidence interval and if the ground truth value for the prediction is within this interval. Table 1 shows this result for four different test instances and five different dropout rates for $T = 100$. Note that when $p = 0.1$, the confidence intervals are smaller but none of the ground truth values were within the respective intervals. As p increases, the standard deviation of the predictions also increases, leading to wider confidence intervals that include the ground truth value. Note that when $p = 0.9$, the ground truth of all four test instances are within their confidence intervals. This behavior is consistent with the interpretation of dropout as a stochastic sampling mechanism over weights, approximating Bayesian inference in neural networks [Gal and Ghahramani 2016, Blundell et al. 2015].

This behavior is further illustrated in Figures 1(a), 1(c) and 1(e), which show the average standard deviation of the test set and the average confidence interval coverage of the ground truth as p increases and for different values of $T \in \{25, 100, 1000\}$. As p increases, both metrics also increase (for all T), however average coverage grows faster than the average standard deviation, showing an important compromise between the uncertainty estimates (standard deviation) and reliability of the predictions (coverage). As p increases to larger values, average coverage reaches 1 but average standard deviations also increase significantly.

Interestingly, as T increases the estimation of uncertainty becomes more stable, and the empirical coverage improves, especially for intermediate dropout rates (around 0.4 - 0.6). For example, fixing $p = 0.5$ and increasing T improves the average coverage with only a small increase in the average standard deviation.

The relationship between p and the estimated uncertainty (standard deviation) is also presented in Figure 2, which illustrate the model's behavior across all test samples sorted by actual house value, under different dropout rates. Each plot shows the deterministic prediction (blue line), the average MC Dropout prediction (green line), the ground truth (black line), and the uncertainty band corresponding to the ± 1 standard deviation interval (shaded area in green).

Dropout	Index	GT	MC Mean	Std. Dev.	95% CI	GT in CI?
0.1	734	$3.98 \cdot 10^4$	$1.81 \cdot 10^5$	$2.46 \cdot 10^4$	$[1.32 \cdot 10^5,\ 2.29 \cdot 10^5]$	No
	950	$1.78 \cdot 10^5$	$1.11 \cdot 10^5$	$9.55 \cdot 10^3$	$[9.21 \cdot 10^4,\ 1.30 \cdot 10^5]$	No
	2425	$1.78 \cdot 10^5$	$1.21 \cdot 10^5$	$8.38 \cdot 10^3$	$[1.05 \cdot 10^5,\ 1.38 \cdot 10^5]$	No
	2992	$5.00 \cdot 10^5$	$2.10 \cdot 10^5$	$2.89 \cdot 10^4$	$[1.53 \cdot 10^5,\ 2.67 \cdot 10^5]$	No
0.2	734	$3.98 \cdot 10^4$	$1.76 \cdot 10^5$	$3.46 \cdot 10^4$	$[1.08 \cdot 10^5,\ 2.43 \cdot 10^5]$	No
	950	$1.78 \cdot 10^5$	$1.22 \cdot 10^5$	$1.44 \cdot 10^4$	$[9.34 \cdot 10^4,\ 1.50 \cdot 10^5]$	No
	2425	$1.78 \cdot 10^5$	$1.28 \cdot 10^5$	$1.32 \cdot 10^4$	$[1.02 \cdot 10^5,\ 1.54 \cdot 10^5]$	No
	2992	$5.00 \cdot 10^5$	$2.27 \cdot 10^5$	$4.03 \cdot 10^4$	$[1.48 \cdot 10^5,\ 3.06 \cdot 10^5]$	No
0.5	734	$3.98 \cdot 10^4$	$1.56 \cdot 10^5$	$4.67 \cdot 10^4$	$[6.49 \cdot 10^4,\ 2.48 \cdot 10^5]$	No
	950	$1.78 \cdot 10^5$	$1.58 \cdot 10^5$	$3.33 \cdot 10^4$	$[9.26 \cdot 10^4,\ 2.23 \cdot 10^5]$	Yes
	2425	$1.78 \cdot 10^5$	$1.47 \cdot 10^5$	$3.21 \cdot 10^4$	$[8.42 \cdot 10^4,\ 2.10 \cdot 10^5]$	Yes
	2992	$5.00 \cdot 10^5$	$2.66 \cdot 10^5$	$6.52 \cdot 10^4$	$[1.38 \cdot 10^5,\ 3.94 \cdot 10^5]$	No
0.7	734	$3.98 \cdot 10^4$	$1.51 \cdot 10^5$	$6.74 \cdot 10^4$	$[1.88 \cdot 10^4,\ 2.83 \cdot 10^5]$	Yes
	950	$1.78 \cdot 10^5$	$1.86 \cdot 10^5$	$6.17 \cdot 10^4$	$[6.48 \cdot 10^4,\ 3.07 \cdot 10^5]$	Yes
	2425	$1.78 \cdot 10^5$	$1.70 \cdot 10^5$	$5.72 \cdot 10^4$	$[5.79 \cdot 10^4,\ 2.82 \cdot 10^5]$	Yes
	2992	$5.00 \cdot 10^5$	$3.17 \cdot 10^5$	$1.28 \cdot 10^5$	$[6.66 \cdot 10^4,\ 5.67 \cdot 10^5]$	Yes
0.8	734	$3.98 \cdot 10^4$	$1.72 \cdot 10^5$	$7.37 \cdot 10^4$	$[2.75 \cdot 10^4,\ 3.16 \cdot 10^5]$	Yes
	950	$1.78 \cdot 10^5$	$2.14 \cdot 10^5$	$1.20 \cdot 10^5$	$[-2.07 \cdot 10^4,\ 4.49 \cdot 10^5]$	Yes
	2425	$1.78 \cdot 10^5$	$1.97 \cdot 10^5$	$1.01 \cdot 10^5$	$[-1.05 \cdot 10^3,\ 3.94 \cdot 10^5]$	Yes
	2992	$5.00 \cdot 10^5$	$2.96 \cdot 10^5$	$1.56 \cdot 10^5$	$[-9.39 \cdot 10^3,\ 6.00 \cdot 10^5]$	Yes

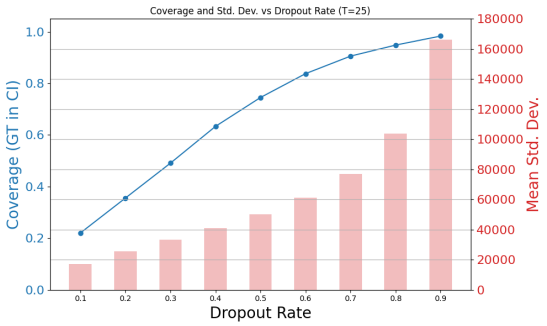
Table 1. Comparison between deterministic predictions and average prediction for different dropout rates. GT = Ground Truth, Std. Dev. = Standard Deviation, CI = Confidence Interval, and "GT in CI?" indicates whether the ground truth lies within the 95% confidence interval defined by $\mu \pm 1.96 \cdot \sigma$.

As the dropout rate increases, there is a clear expansion of the uncertainty band, reflecting the growth in the standard deviation of the stochastic predictions. This supports the interpretation of MC Dropout as an approximate Bayesian inference method, where higher dropout rates induce greater variability among sampled submodels. On the other hand, the average prediction tends to diverge from the deterministic prediction (and eventually from the ground truth) as p increases, suggesting degradation in predictive accuracy.

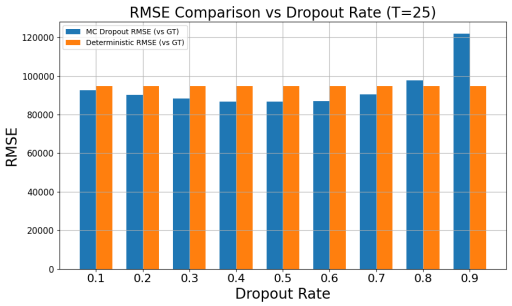
For instance, in Figure 2(a) (dropout 0.1), the uncertainty bands are narrow, and the MC mean closely matches the deterministic prediction. In contrast, Figure 2(e) (dropout 0.9) shows such a wide uncertainty band that the $\pm 1\sigma$ interval spans an extremely large range, with evidence of the model overestimating uncertainty even in regions of low variability. This behavior reinforces the need to calibrate the dropout rate p used in inference to avoid both underestimation and overestimation of predictive uncertainty.

4.2. Error (RMSE)

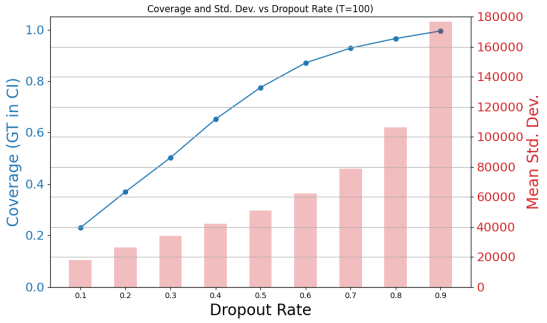
The second analysis considers the error (RMSE) of the average prediction with respect to the ground truth as a function of p. Figure 1(b), 1(d), and 1(f) show the RMSE of



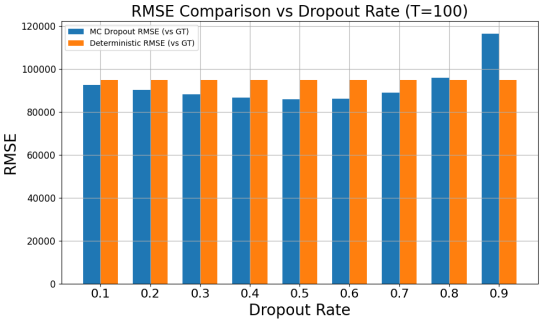
(a) $T = 25$: CI Coverage and Std. Deviation



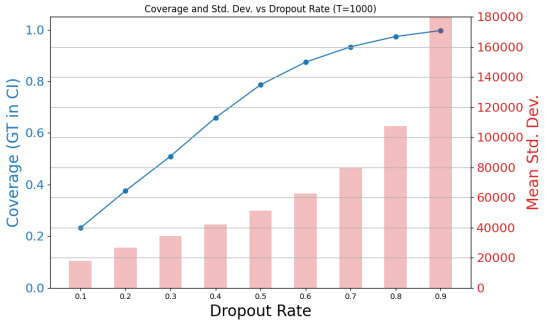
(b) $T = 25$: RMSE



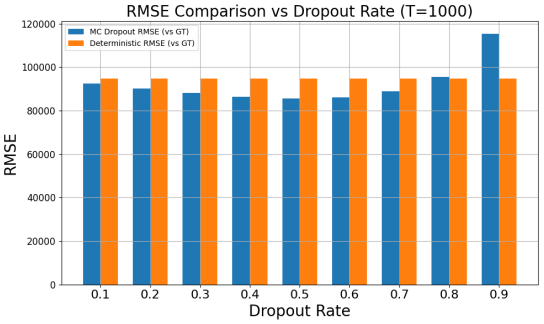
(c) $T = 100$: CI Coverage and Std. Deviation



(d) $T = 100$: RMSE



(e) $T = 1000$: CI Coverage and Std. Deviation



(f) $T = 1000$: RMSE

Figure 1. Average standard deviation, confidence interval coverage and RMSE for different dropout rates p and number of samples T.

the test set as a function of p for different values of T. The RMSE of the deterministic prediction is also shown for comparison (single prediction, without applying any dropout at inference time). Interestingly, the results show a non-monotonic behavior in RMSE with the smallest errors occurring at moderate dropout values (around 0.4 - 0.6). As p increases beyond 0.6, the error increases and eventually becomes larger than the error for the deterministic prediction. This non-monotonic behavior is observed for all T values considered.

Interestingly, the larger coverage of the ground truth for p values above 0.6 comes with a larger prediction error (larger RMSE). This indicates an important tradeoff between prediction error and prediction confidence and is related to *miscalibration*, a common challenge in variational methods [Abdar et al. 2021, Kendall and Gal 2017].

4.3. Calibration and Sensitivity

The results suggest that uncertainty estimation via MC Dropout is highly sensitive to the dropout rate used during inference, supporting the hypothesis that this hyperparameter directly affects the magnitude of epistemic uncertainty that is captured by the model. Works such as [Gal and Ghahramani 2016, Mukhoti et al. 2020] emphasize the importance of calibrating confidence intervals, particularly when a probabilistic interpretation is desired. The variability in confidence interval width for extreme ground truth values (e.g., close to \$500,000) when p is large further highlights this point. For such large ground truth values, confidence interval width exceeds \$800,000 in some cases. See largest values in Figure 2.

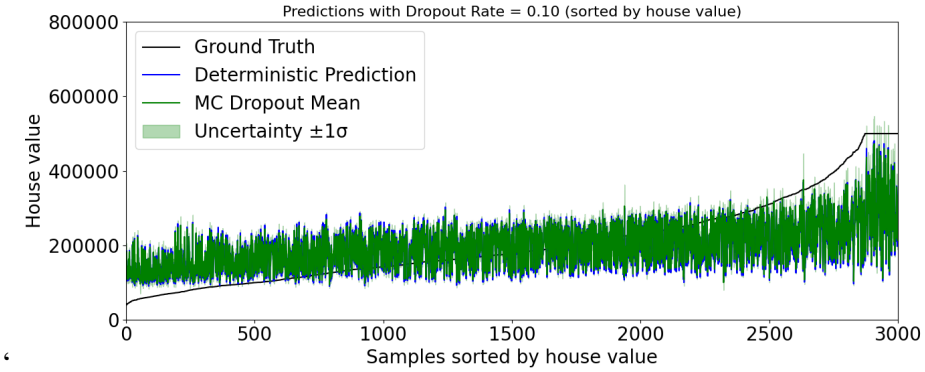
4.4. Practical Implications

Our analysis shows that while MC Dropout is computationally efficient, it requires careful selection of the inference dropout rate, especially when used as a tool for uncertainty quantification. In critical settings, adopting calibration criteria based on cross-validation or specific metrics such as the *Expected Calibration Error (ECE)* [Mukhoti et al. 2020] may help mitigate arbitrary choices of this hyperparameter. In summary, the flexibility of tuning the dropout rate during inference allows modulating the model's sensitivity, but its influence on the quality of uncertainty estimation should not be overlooked.

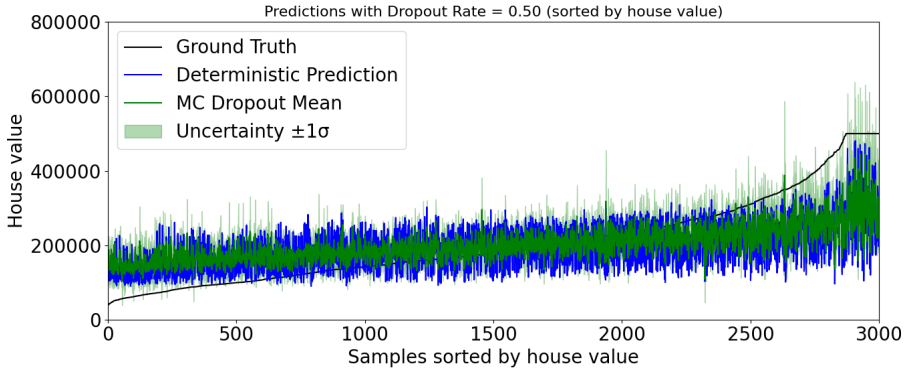
5. Conclusion

This work investigated the impact of the dropout rate during inference on the estimation of epistemic uncertainty in deep neural networks. Using the MC Dropout method, we conducted systematic experiments varying the dropout rate from 0.1 to 0.9, while keeping the model fixed after training with a dropout rate of 0.5. The results revealed that the stochastic variability induced by different rates strongly influences the magnitude of the predictive standard deviation and, consequently, the width of the confidence intervals. This has a direct consequence on the coverage of the ground truth values by the confidence interval. Interestingly, results indicated that the error (RMSE) of the average prediction has a non-monotonic behavior with smaller errors between 0.4 and 0.6, indicating a trade-off between prediction error and prediction confidence.

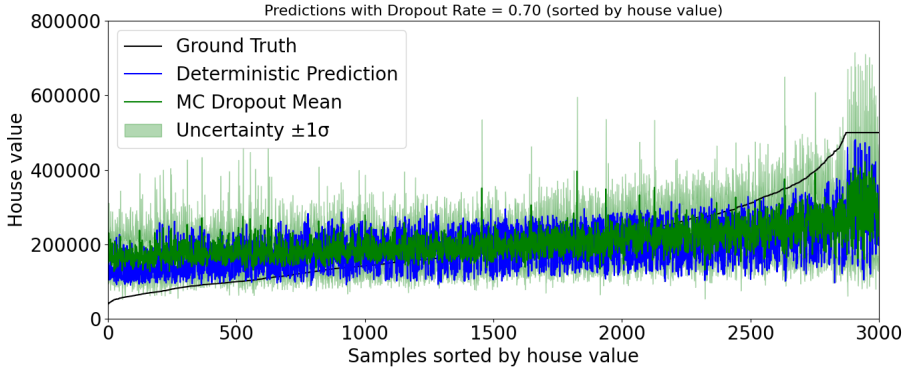
The findings in this work indicate that the dropout rate should not be treated solely as a training hyperparameter, but also as a critical control factor for predictive confidence



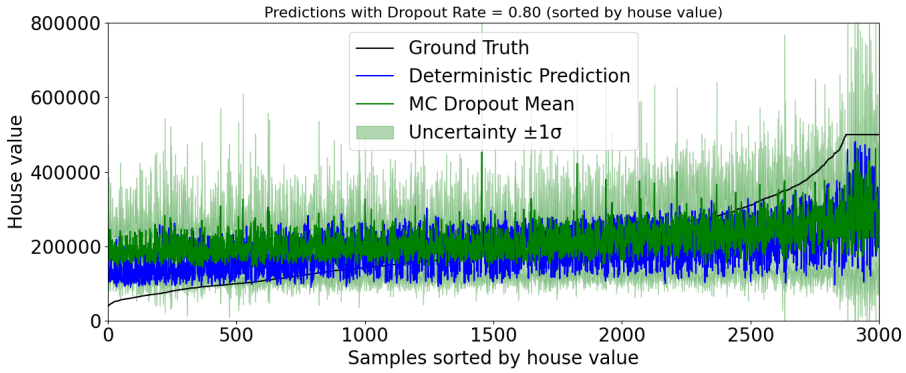
(a) Dropout = 0.1



(b) Dropout = 0.5



(c) Dropout = 0.8



(d) Dropout = 0.9

Figure 2. Average prediction, standard deviation and ground truth in the test set for different dropout rates.

during inference. This aspect must be carefully considered in applications where uncertainty quantification is essential for decision-making.

Acknowledgments This work received financial support through research grants from CNPq (310742/2023-4), FAPERJ (E-26/200.483/2023 and E-26/202.517/2024) and CAPES (PROEX).

References

Abdar, M., Pourpanah, F., Hussain, S., Rezazadegan, D., et al. (2021). A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*.

Baldi, P. and Sadowski, P. J. (2013). Understanding dropout. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. (2015). Weight uncertainty in neural networks. In *International Conference on Machine Learning*.

Der Kiureghian, A. and Ditlevsen, O. (2009). Aleatory or epistemic? does it matter? *Structural Safety*.

Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M., and Thrun, S. (2017). Dermatologist-level classification of skin cancer with deep neural networks. *Nature*.

Gal, Y. and Ghahramani, Z. (2016). Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning (ICML)*.

Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D., and Wilson, A. G. (2018). Averaging weights leads to wider optima and better generalization. In *Proceedings of the 34th Conference on Uncertainty in Artificial Intelligence (UAI)*.

Kendall, A. and Gal, Y. (2017). What uncertainties do we need in bayesian deep learning for computer vision? *Neural Information Processing Systems (NeurIPS)*.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in Neural Information Processing Systems (NeurIPS)*.

Louizos, C., Kingma, D. P., and Welling, M. (2017). Multiplicative normalizing flows for variational bayesian neural networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*.

Mukhoti, J., Ashukha, A., Lampert, C. H., and Gal, Y. (2020). Calibrating deep neural networks using focal loss. In *Advances in Neural Information Processing Systems*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*.

Walker, W. E., Harremöes, P., Rotmans, J., van der Sluijs, J. P., van Asselt, M. B. A., Janssen, P., and Krayer von Krauss, M. P. (2003). Defining uncertainty: a conceptual basis for uncertainty management in model-based decision support. *Integrated Assessment*.