
Unified Uncertainties: Combining Input, Data and Model Uncertainty into a Single Formulation

Matias Valdenegro-Toro¹ Ivo Pascal de Jong¹ Marco Zullo¹

Abstract

Modelling uncertainty in Machine Learning models is essential for achieving safe and reliable predictions. Most research on uncertainty focuses on output uncertainty (predictions), but minimal attention is paid to uncertainty at inputs. We propose a method for propagating uncertainty in the inputs through a Neural Network that is simultaneously able to estimate input, data, and model uncertainty. Our results show that this propagation of input uncertainty results in a more stable decision boundary even under large amounts of input noise than comparatively simple Monte Carlo sampling. Additionally, we discuss and demonstrate that input uncertainty, when propagated through the model, results in model uncertainty at the outputs. The explicit incorporation of input uncertainty may be beneficial in situations where the amount of input uncertainty is known, though good datasets for this are still needed.

1. Introduction

While neural networks are state of the art for many problems ranging from language generation to image interpretation, their predictions are often overconfident and they have issues estimating their own uncertainty (Valdenegro-Toro & Mori, 2022; Ovadia et al., 2019).

One often overlooked source of uncertainty is the model’s input (Tzelepis et al., 2017), as features and pixel values can be noisy, depending on the data source. Uncertainty in the input is an underexplored research area (Rodrigues et al., 2021; Hüllermeier, 2014; Depeweg et al., 2018), with most works about uncertainty estimation being about output uncertainty (Hüllermeier & Waegeman, 2021). Accounting

¹Department of Artificial Intelligence, University of Groningen, Groningen, The Netherlands. Correspondence to: Matias Valdenegro-Toro <m.a.valdenegro.toro@rug.nl>.

LatinX in AI Research Workshop at the *International Conference on Machine Learning* 2024, Vienna, Austria. Copyright 2024 by the author(s).

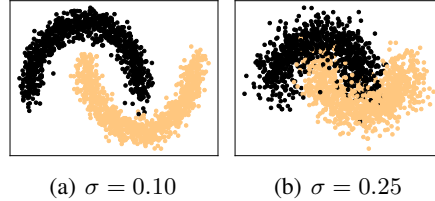


Figure 1. Concept of Input Uncertainty in the Two Moons dataset, increasing σ makes classification more difficult and decision boundary unclear. Considering IU could improve model performance.

for the uncertainty in the input can improve the final prediction, as well as its uncertainty. Appendix B offers further discussion on related work.

In this work we introduce the concept of input uncertainty, as well as a new formulation for simultaneously estimating data uncertainty (aleatoric—AU), model uncertainty (epistemic—EU) and input uncertainty—IU. Additionally, we argue that propagation of IU into the model results in epistemic uncertainty.

The contributions of this work are: (a) a novel formulation combining AU, EU, and IU into a single model and uncertainty estimation method, (b) a theoretically grounded interpretation that IU is processed as EU, and (c) an experimental validation of our formulation.

2. Input Uncertainty Formulation

Notation. We denote standard supervised learning datasets as $(\mathbf{x}_j, y_j)_{j=1}^N$. Datasets can have IU, which is denoted as $\mathbf{x} = \{\mu^i, \sigma^i\}$ with i denoting input variables. Datasets can additionally have output uncertainty, which is denoted differently for regression and classification. For regression $y = \{\mu^o, \sigma^o\}$, with o denoting output variables. For classification $y = P(\hat{y}=c | \mathbf{x}) \quad \forall c \in \{1, \dots, C\}$, where probabilities can be obtained via sampling through the softmax function $P(\hat{y}=c | \mathbf{x}) \sim \text{softmax}(z_c)$ and $z_c \sim \mathcal{N}(\hat{\mu}^o, \hat{\sigma}^o)$. A model is a function f_θ with trainable parameters θ . Ground truth labels are denoted as y , while predicted values are \hat{y} .

We define the input/output uncertainty estimation problem

as training a model f_θ where both IUs σ^i and output uncertainties σ^o are considered, and related by:

$$\hat{y}^o = f_\theta(\mathbf{x}^i) \quad (1)$$

A common desideratum is that output uncertainties are calibrated. To denote inputs and outputs with uncertainty, we use the notation $\mu \pm \sigma$, where μ is the prediction or input value and σ is its associated uncertainty, represented by the standard deviation.

IU Propagation. Given an input with uncertainty $x = \{\mu^i, \sigma^i\}$, it is possible to propagate the uncertainties through a model f_θ by using a first order Taylor approximation (Kelly, 1994):

$$f_\theta(\mu^i \pm \sigma^{2i}) \approx f_\theta(\mu) \pm \mathbf{J}\sigma^2\mathbf{J}^T \quad (2)$$

Where $\mu^o = f_\theta(\mu^i)$ and $\sigma^{o2} = \mathbf{J}\sigma^{i2}\mathbf{J}^T$ and the Jacobian matrix is $\mathbf{J}_{ij} = \frac{\partial f_i}{\partial \mu_j}$ evaluated at μ^i .

IU MC Sampling. An alternative way to propagate IU to the output of a model f_θ is via Monte Carlo sampling:

$$f_\theta(\mu^i \pm \sigma^{2i}) \approx \mathbb{E}[f_\theta(\tilde{\mathbf{x}})] \pm \text{Var}[f_\theta(\tilde{\mathbf{x}})] \quad \tilde{\mathbf{x}} \sim \mathcal{N}(\mu^i, \sigma^{2i}) \quad (3)$$

The input is modelled as Gaussian random variable $\tilde{\mathbf{x}}$, from which samples are drawn that are passed through the model f_θ . The output is an approximate Gaussian $\mathcal{N}(f_\theta(\tilde{x}), \text{Var}[f_\theta(\tilde{x})])$. The monte carlo approximation is computed using N samples:

$$\begin{aligned} \mathbb{E}[f_\theta(\tilde{\mathbf{x}})] &\approx N^{-1} \sum_{j=1}^N f_\theta(\tilde{\mathbf{x}}_j) \\ \text{Var}[f_\theta(\tilde{\mathbf{x}})] &\approx (N-1)^{-1} \sum_{j=1}^N (f_\theta(\tilde{\mathbf{x}}_j) - \mathbb{E}[f_\theta(\tilde{\mathbf{x}})])^2 \end{aligned}$$

With $\tilde{\mathbf{x}}_j$ being the j -th sample from $\mathcal{N}(\mu^i, \sigma^{2i})$. Both approaches are approximate, as there is no analytic formulation for an exact propagation of uncertainty.

2.1. Uncertainty Estimation Formulation

In this section we combine IU with standard formulations for AU and EU. For IU estimation, we assume the existence of a function `propagate`(μ, σ^2) that propagates IU through the model, as described in the previous section, and `epistemic`(μ) corresponds to standard uncertainty estimation methods that produce EU via sampling (e.g. MC-Dropout (Gal & Ghahramani, 2016), MC-DropConnect (Mobiny et al., 2021), Bayesian Neural Networks like Flipout (Wen et al., 2018), etc) or ensembles (e.g. Deep Ensembles (Lakshminarayanan et al., 2017)). We combine these by first applying the EU estimation method, and then

the IU propagation method:

$$\begin{aligned} S_{\text{epi}}(\mu, \sigma^2) &= \text{propagate}(\text{epistemic}(\mu), \sigma^2) \\ \mu^s &= \mathbb{E}[S_{\text{epi}}(\mu, \sigma^2)] \\ \sigma^{2s} &= \text{Var}[S_{\text{epi}}(\mu, \sigma^2)] \end{aligned}$$

Where $S_{\text{epi}}(\mu, \sigma^2)$ represents the result of the IU propagation method, corresponding to N samples (on which expectation and variance are computed), depending on the IU propagation method, while the EU method uses M forward passes. Then three values can be estimated:

$$\begin{aligned} \mu^o &= \mathbb{E}[\mu^s] && \text{(Prediction)} \\ \sigma_{\text{inp}}^o &= \text{Var}[\mu^s] && \text{(IU)} \\ \sigma_{\text{epi}}^o &= \mathbb{E}[\sigma^{2s}] && \text{(EU)} \end{aligned}$$

In general terms, μ^o is the model prediction after averaging over the EU estimation method, σ_{inp} is the output uncertainty attributed to the input, being the disagreement or variation between the samples, and σ_{epi} is the output EU, given by the average of the per-input-sample EUs produced by `epistemic`(μ).

Depending on the task, the prediction also holds AU.

Classification. For classification with C classes, output uncertainties ($\mu^o, \sigma_{\text{inp}}, \sigma_{\text{epi}}$) are modelled as logits (values before applying softmax), so they are C -dimensional vectors that can be transformed into probabilities by:

$$\begin{aligned} p_{\text{ale}}(y | \mathbf{x}) &= \text{softmax}(\mu^o) && \text{(AU)} \\ p_{\text{inp}}(y | \mathbf{x}) &= \text{sampling_softmax}(\mu^o, \sigma_{\text{inp}}) && \text{(IU)} \\ p_{\text{epi}}(y | \mathbf{x}) &= \text{sampling_softmax}(\mu^o, \sigma_{\text{epi}}) && \text{(EU)} \end{aligned}$$

Where the sampling softmax function (Valdenegro-Toro & Mori, 2022) is defined by:

$$\begin{aligned} \text{sampling_softmax}(\mu(\mathbf{x}), \sigma^2(\mathbf{x})) &= N^{-1} \sum_j \text{softmax}(\hat{\mathbf{z}}_j) \\ &\text{with } \hat{\mathbf{z}}_j \sim \mathcal{N}(\mu(\mathbf{x}), \sigma^2(\mathbf{x})) \quad (4) \end{aligned}$$

Where N is the number of samples used for Monte Carlo sampling.

Regression. Regression requires small changes to the formulation. To estimate AU in regression, a model with two output heads is required, and the Gaussian negative log-likelihood loss (Seitzer et al., 2022a) is used to train both mean μ and variance σ^2 heads.

$$L_{NLL}(y_j, \mathbf{x}_j) = \log \sigma^2(\mathbf{x}_j) + \frac{(\mu(\mathbf{x}_j) - y_j)^2}{\sigma^2(\mathbf{x}_j)}. \quad (5)$$

The model's forward passes now produce mean and variance, implying that `epistemic`(μ) now returns two values. Some adjustments are needed for the regression formulation, to propagate aleatoric variances from the model

forward passes through the epistemic and IU propagation methods, without changes, only producing samples.

$$S_{\text{epi}}(\mu, \sigma^2), \sigma_{\text{ale}}^2 = \text{propagate}(\text{epistemic}(\mu), \sigma^2) \quad (6)$$

Then the associated predictions and uncertainties for regression are:

$$\mu^o, \sigma_{\text{ale}}^o = \mathbb{E}[\mu^s], \mathbb{E}[\sigma_{\text{ale}}^2] \quad (\text{Prediction \& AU})$$

$$\sigma_{\text{inp}}^o = \text{Var}[\mu^s] \quad (\text{IU})$$

$$\sigma_{\text{epi}}^o = \mathbb{E}[\sigma^{2s}] \quad (\text{EU})$$

The output aleatoric variance σ_{ale}^o is the expectation over aleatoric uncertainty samples propagated through both `propagate()` and `epistemic()`.

2.2. Discussion

Uncertainty Interpretation. Conceptually the IU σ^i is part of AU, as its source is the training data and the sensor or device producing the original data. IU does not reduce if more information or data is added, as it is inherent to the data.

When propagating IU through a model, the output uncertainties σ_{inp}^o and σ_{epi}^o are EUs, as they correspond to uncertainty due to the model’s equations, and it is combined with an EU estimation method. This is a novel view as we argue that input AU is transformed into output EU via our proposed uncertainty formulation. The predictive uncertainty can be conceptually combined as:

$$\text{Predictive Uncertainty} = \text{Aleatoric} + \text{Epistemic} + \text{Input} \quad (7)$$

Computational Costs. Methods for IU estimation have increased computational requirements during inference. For IU propagation using a Taylor approximation, the cost is $\mathcal{O}(M)$, as only EU estimation contribute to increased costs, but sampling IU estimation has cost $\mathcal{O}(NM)$, due to N samples for IU propagation, and each of them needs M samples/ensembles for EU estimation. This makes sampling infeasible on large models and datasets, while using the Taylor approximation incurs minimal additional cost.

3. Experiments

Experimental Setup. For experimentation we use the two moons dataset because it is simple, we can add noise to input data points to simulate uncertainty, and it allows a clear visualization of uncertain regions and the decision boundary. We train a four layer multi-layer perceptron using ReLU activations, the two last layers use some form of uncertainty estimation method as described below. Models are trained with the Adam optimizer using the cross-entropy loss.

Gaussian distributed noise with $\mu = 0$ and $\sigma = 0.1$ is added to the training set, and for IU propagation evaluation, we use increasing levels of IU: $\sigma \in \{0.25, 0.5, 1.0\}$.

EU Methods. We evaluate a standard neural network (without EU methods), MC-Dropout (Gal & Ghahramani, 2016) with drop probability $p = 0.2$, MC-DropConnect (Mobiny et al., 2021) with drop probability $p = 0.05$, Ensembles (Lakshminarayanan et al., 2017) with $M = 5$ ensemble members, and Flipout (Wen et al., 2018). Stochastic methods use $M = 20$ forward passes.

Expectations. Validating IU propagation is difficult as there is no ground truth label for uncertainties, and novelty in formulations require a different evaluation approach. To evaluate the quality of IU propagation, we train a model with a certain level of IU on training data, and test model predictions and associated uncertainties with increasing levels of IU, expecting that the model will output increasing uncertainties, in particular σ_{inp}^o should increase together with the IU, while AUs should remain approximately constant. EU should also increase but not at the same level as σ_{inp}^o .

3.1. Classification on Two Moons Dataset with IU

Figures 2 and 3 demonstrate predicted AU, EU, and IU under different amounts of IU using respectively the Monte Carlo Sampling approach and the Uncertainty Propagation approach across different methods for quantifying EU. The most salient difference between these two approaches, which seems consistent regardless of the method for quantifying EU, is that the Propagation approach maintains a clear decision boundary under increased input noise, while the Monte Carlo Sampling approach shows a decision boundary that becomes noisy and loses shape under high amounts of input noise. This suggests propagation of input uncertainty may affect the decision boundary.

The second difference is that under the Uncertainty Propagation approach, the predicted IU changes very little, but the predicted EU increases. This effect also exists in the Monte Carlo approach, but is less pronounced. The propagation of IU leads to an increase in EU as it offers the necessary information so that a better decision boundary may be learned, but this does add complexity to the learning problem.

Overall, the Monte Carlo Sampling shows that the increase in IU adds noise to all predicted uncertainties. However, for the Uncertainty Propagation, only the EU changes. This shows that the Uncertainty Propagation approach allows for a more consistent disentanglement of the different kinds of uncertainty, and allows the preservation of decision boundaries even under large amounts of input noise.

The main drawback of the Uncertainty Propagation approach is that it fails to predict an increase in IU even though the ground truth IU does increase. The Sampling IU does predict an increase in IU.

The different methods for predicting EU exhibit similar behaviour, showing that IU has a minor role in direct epistemic

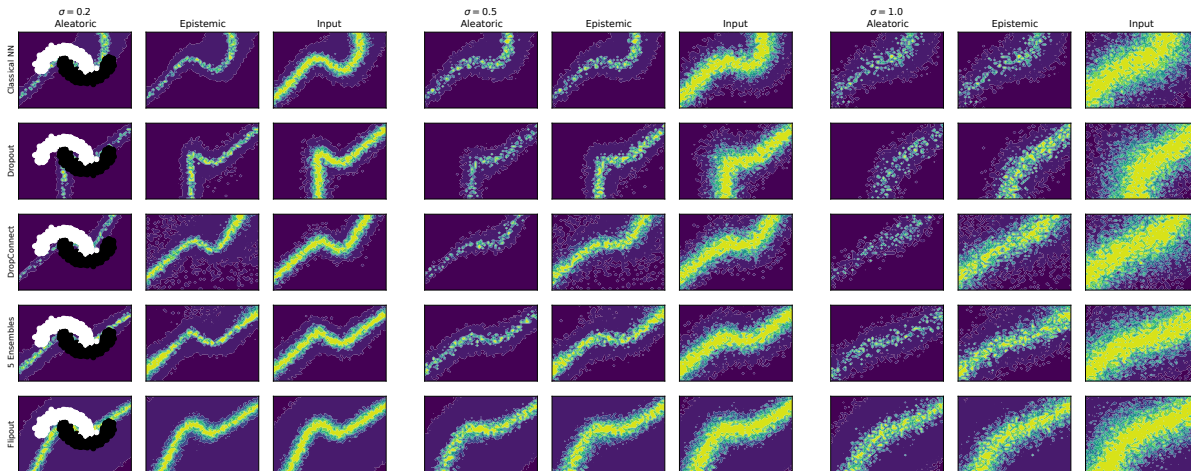


Figure 2. Results on the Two Moons dataset with Monte Carlo Sampling IU (Using Eq 3). As the IU increases, the learned decision boundary loses shape and becomes noisy. Note that each value of σ produces three plot columns for each output uncertainty.

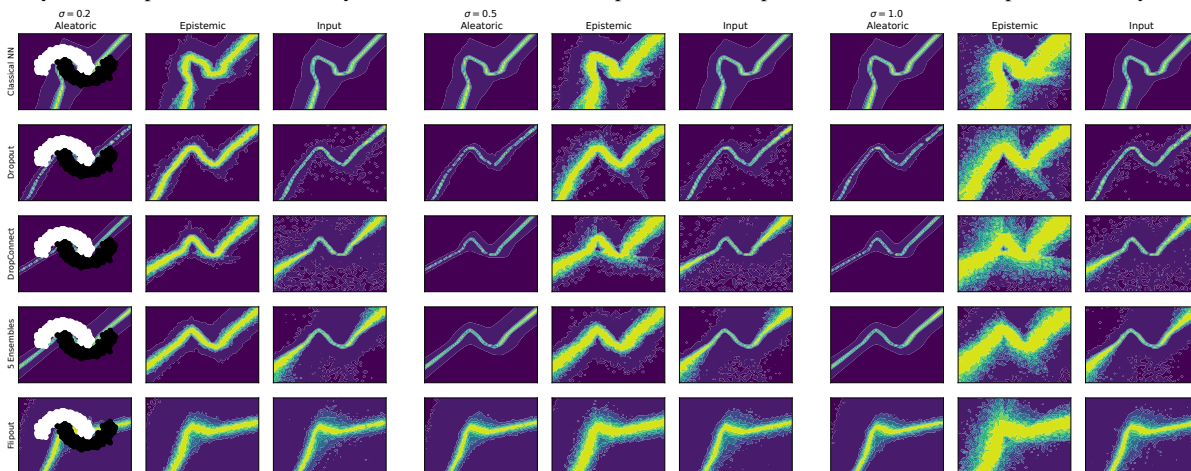


Figure 3. Results on the Two Moons dataset with Propagation IU (Using Eq 2). As the IU increases, the predicted EU increases. The other predicted uncertainties remain roughly the same. Note that each value of σ produces three plot columns for each output uncertainty.

uncertainty, clearly being reflected via the method used to propagate IU through the model. Appendix C shows that the findings for classification also hold for regression.

4. Conclusions and Future Work

In this work we proposed a new formulation for unified uncertainties considering aleatoric, epistemic, and input uncertainty. This allows for input uncertainty to be explicitly modelled in a neural network.

The results show that the proposed Uncertainty Propagation method results in better uncertainty estimates and more consistent decision boundaries than a comparatively simple baseline method of Monte Carlo Sampling from the input uncertainty. Input uncertainty is often trivialised as noise that can just be learned with sufficient samples. Our work shows that knowledge about the input uncertainty may be

propagated effectively in a Neural Network resulting in more desirable predictive behaviour of the model. Additionally, we show a new insight that this aleatoric input uncertainty becomes epistemic uncertainty in the output if propagated through the model.

The experiments with toy data and the underlying theory show that this is a promising new avenue for further research on uncertainty quantification. To really capitalise on input uncertainty these methods should be applied to cases where input uncertainty may be estimated from prior knowledge. Depth-sensing cameras such as Kinect for example are known to have a higher depth-variance on object further away, and many sensors are manufactured with known amounts of (im)precision in their observation. Collecting this kind of information in datasets is the next step in exploring the viability of propagating input uncertainty through a Neural Network.

References

- Bócsi, B. A. and Csató, L. Hessian corrected input noise models. In *International Conference on Artificial Neural Networks*, pp. 1–8. Springer, 2013.
- Depeweg, S., Hernandez-Lobato, J.-M., Doshi-Velez, F., and Udluft, S. Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning. In *International Conference on Machine Learning*, pp. 1184–1193. PMLR, 2018.
- Gal, Y. and Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059. PMLR, 2016.
- Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. In *International conference on machine learning*, pp. 1321–1330. PMLR, 2017.
- Hüllermeier, E. Learning from imprecise and fuzzy observations: Data disambiguation through generalized loss minimization. *International Journal of Approximate Reasoning*, 55(7):1519–1534, 2014.
- Hüllermeier, E. and Waegeman, W. Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. *Machine Learning*, 110(3):457–506, 2021.
- Kelly, A. *A 3D state space formulation of a navigation Kalman filter for autonomous vehicles*. Carnegie Mellon University, the Robotics Institute, 1994.
- Lakshminarayanan, B., Pritzel, A., and Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.
- Matsuoka, K. Noise injection into inputs in back-propagation learning. *IEEE Transactions on Systems, Man, and Cybernetics*, 22(3):436–440, 1992.
- Mobiny, A., Yuan, P., Moulik, S. K., Garg, N., Wu, C. C., and Van Nguyen, H. Dropconnect is effective in modeling uncertainty of bayesian deep networks. *Scientific reports*, 11(1):1–14, 2021.
- Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J., Lakshminarayanan, B., and Snoek, J. Can you trust your model’s uncertainty? evaluating predictive uncertainty under dataset shift. *Advances in neural information processing systems*, 32, 2019.
- Rodrigues, N. V., Abramo, L. R., and Hirata, N. S. The information of attribute uncertainties: what convolutional neural networks can learn about errors in input data. *arXiv preprint arXiv:2108.04742*, 2021.
- Seitzer, M., Tavakoli, A., Antic, D., and Martius, G. On the pitfalls of heteroscedastic uncertainty estimation with probabilistic neural networks. In *International Conference on Learning Representations*, 2022a. URL <https://openreview.net/forum?id=aPOpXlnV1T>.
- Seitzer, M., Tavakoli, A., Antic, D., and Martius, G. On the pitfalls of heteroscedastic uncertainty estimation with probabilistic neural networks. *arXiv preprint arXiv:2203.09168*, 2022b.
- Sietsma, J. and Dow, R. J. Creating artificial neural networks that generalize. *Neural networks*, 4(1):67–79, 1991.
- Tzelepis, C., Mezaris, V., and Patras, I. Linear maximum margin classifier for learning from uncertain data. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2948–2962, 2017.
- Valdenegro-Toro, M. and Mori, D. S. A deeper look into aleatoric and epistemic uncertainty disentanglement. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 1508–1516. IEEE, 2022.
- Wen, Y., Vicol, P., Ba, J., Tran, D., and Grosse, R. Flipout: Efficient pseudo-independent weight perturbations on mini-batches. *arXiv preprint arXiv:1803.04386*, 2018.
- Wimmer, L., Sale, Y., Hofman, P., Bischl, B., and Hüllermeier, E. Quantifying aleatoric and epistemic uncertainty in machine learning: Are conditional entropy and mutual information appropriate measures? In *Uncertainty in Artificial Intelligence*, pp. 2282–2292. PMLR, 2023.
- Wright, W. Neural network regression with input uncertainty. In *Neural Networks for Signal Processing VIII. Proceedings of the 1998 IEEE Signal Processing Society Workshop (Cat. No. 98TH8378)*, pp. 284–293. IEEE, 1998.
- Wright, W., Ramage, G., Cornford, D., and Nabney, I. T. Neural network modelling with input uncertainty: Theory and application. *Journal of VLSI signal processing systems for signal, image and video technology*, 26:169–188, 2000.

A. Broader Impact Statement

Machine learning models and systems using them are generally not robust to different types of corruptions in the input data, and we believe that explicitly modelling uncertainty and noise in the input data can lead to more robust models, as more information is provided to make a prediction.

But as with any uncertainty estimation method, there are no guarantees about correctness of predictions and their associated uncertainties, so these methods should be used with care and extensive validation.

Some limitations of our work are: we experiment only on toy datasets to understand how methods are actually working, and real-world performance is left for future work. We only test with input Gaussian noise, while real-world corruptions might follow distributions that considerably deviate from Gaussian noise.

B. Related Work

Most existing research on Uncertainty Quantification focuses on AU (Seitzer et al., 2022b), EU (Gal & Ghahramani, 2016; Lakshminarayanan et al., 2017; Wen et al., 2018), or both (Valdenegro-Toro & Mori, 2022; Wimmer et al., 2023), but very little work focuses on IU.

In the typical BNN-based setup AU is generally estimated by a predicted variance for regression (Seitzer et al., 2022b), or by relying on the softmax function for classification (Guo et al., 2017). The desideratum in AU estimation is to have the predicted probability of a prediction match the probability of it being correct (accuracy). This is challenging under heteroscedastic aleatoric uncertainty, where the probability of being correct may change between different samples.

EU is modelled by uncertainty in the parameters. In true Bayesian Neural Networks each parameter in the Neural Network is represented with an arbitrary probability distribution. Learning this is computationally infeasible, so a variety of approximation methods have been proposed with varying estimation quality and practical consequences. MC-Dropout (Gal & Ghahramani, 2016) and MC-Dropconnect (Mobiny et al., 2021) require minimal changes to the model architecture, and do not make any changes to the training process, but are M times as computationally expensive during inference. Deep Ensembles (Lakshminarayanan et al., 2017) train N models on the same data with different random initialization. This puts no requirements on the model architecture, and does not require as many forward passes to get good results, but makes training N times more expensive and requires more memory to store the models. Flipout (Wen et al., 2018) requires changes to the architecture, but does actually learn a distribution for the parameters.

Research on IU is limited and has not been connected with

modern UQ literature. Matsuoka (1992) and Sietsma & Dow (1991) explored injecting uncertainty at the inputs as a form of regularization, but did not consider propagating IU to achieve better estimated uncertainties. Their findings input uncertainty regularizes can also be seen in Figure 2. Wright et al. (2000) and Wright (1998) does propagate IU through a Bayesian Neural Network for regression with the Laplace approximation. They show that the predicted uncertainty increases corresponding with the input uncertainty, as is also shown in our results.

Bócsi & Csató (2013) proposes using the Hessian to correct biases when integrating input noise, corresponding to IU.

C. Regression with IU on Toy Dataset

We additionally produce results on a toy regression problem, to test the equivalency of the regression formulation. We use a synthetic function corrupted with noise, defined by:

$$f(x) = x \sin(x) + \epsilon_1 + \epsilon_2 x \quad (8)$$

With $\epsilon_1 \sim \mathcal{N}(0, 0.3^2)$ and $\epsilon_2 \sim \mathcal{N}(0, 0.3^2)$. The use of both noise terms is to introduce homoscedastic (ϵ_1) and heteroscedastic (ϵ_2) aleatoric uncertainty to the data. No noise is added to the input variable x .

We generate $N = 1000$ equally spaced samples in $x \in [0, 10.0]$ for training the model, and additionally generate $N = 200$ samples in $x \in [10, 12]$ as out of distribution samples, to test behavior of input and epistemic uncertainty in this setting.

Input uncertainty is produced by corrupting the input with Gaussian noise with $\mu=0$ and increasing levels of noise following $\sigma \{0.25, 0.5, 1.0\}$.

Results for both sampling and propagation are presented in Figures 4 and 5 correspondingly. These results show similar behavior as with classification on two moons, with Monte Carlo sampling input uncertainty behaving as expected, while input uncertainty propagation turns input uncertainty into epistemic uncertainty, which is unexpected but in line with classification experiments.

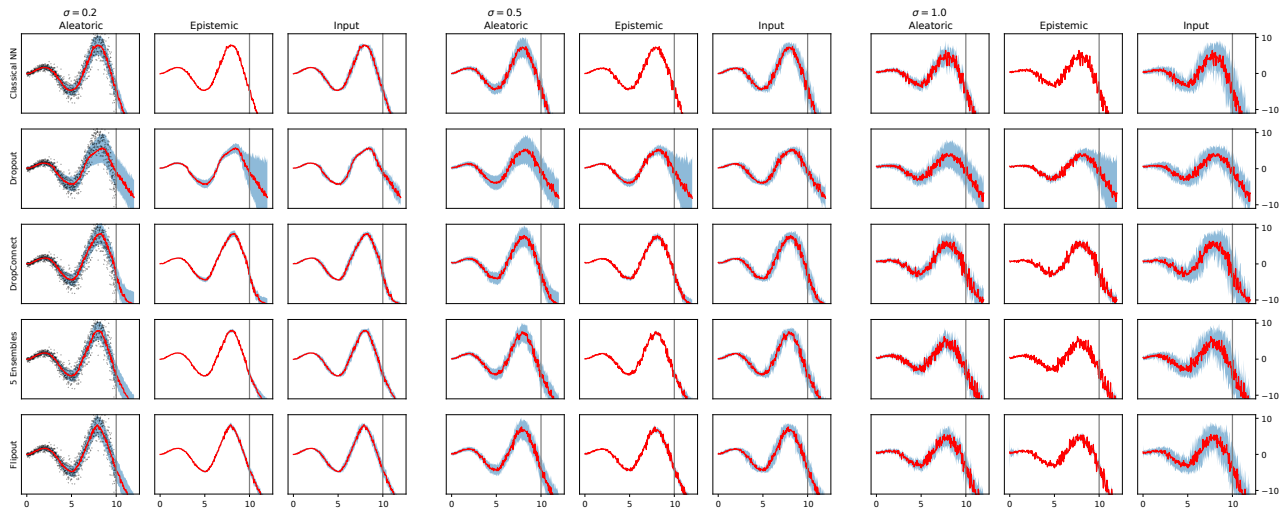


Figure 4. Results on the Toy Regression example with Monte Carlo Sampling IU. As the IU increases, predicted IU increases accordingly, without affecting EU. Note that each value of σ produces three plot columns for each output uncertainty type (Aleatoric, Epistemic, Input).

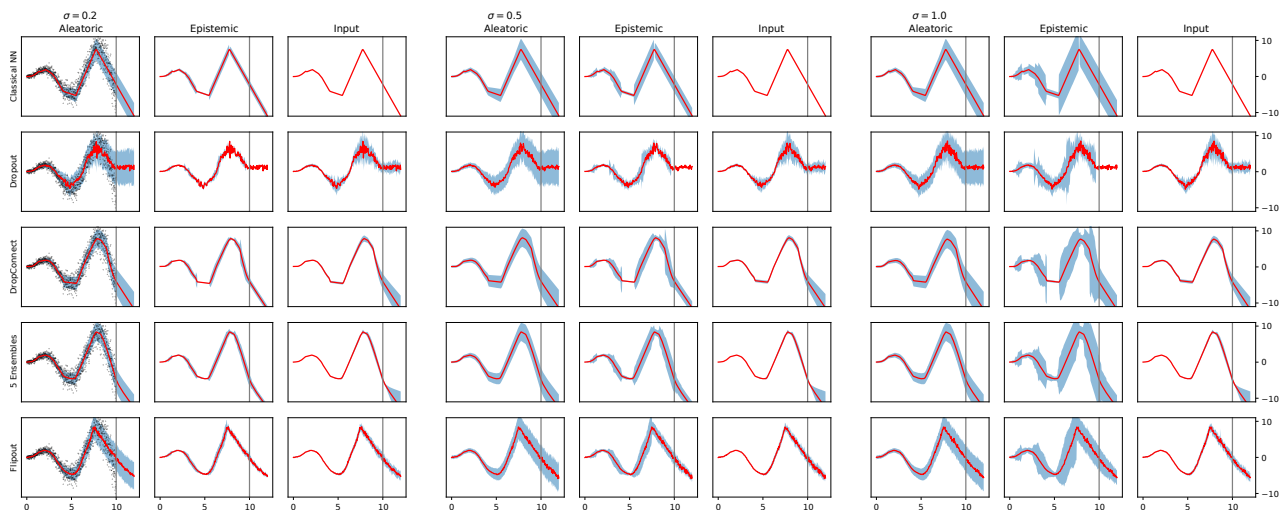


Figure 5. Results on the Toy Regression example with Propagation IU. As the IU increases, mostly predicted EU increases, while predicted IU increases only slightly. Note that each value of σ produces three plot columns for each output uncertainty type (Aleatoric, Epistemic, Input).