UNCERTAINTY AWARE TROPICAL CYCLONE WIND SPEED ESTIMATION FROM SATELLITE DATA

Nils Lehmann
Technical University of Munich
n.lehmann@tum.de

Nina Maria Gottschling
EO Data Science, DLR
nina-maria.gottschling@dlr.de

Stefan Depeweg
Siemens AG
stefan.depeweg@siemens.com

Eric Nalisnick
University of Amsterdam
e.t.nalisnick@uva.nl

ABSTRACT

Deep neural networks (DNNs) have been successfully applied to earth observation (EO) data and opened new research avenues. Despite the theoretical and practical advances of these techniques, DNNs are still considered black box tools and by default are designed to give point predictions. However, the majority of EO applications demand reliable uncertainty estimates that can support practitioners in critical decision making tasks. This work provides a theoretical and quantitative comparison of existing uncertainty quantification methods for DNNs applied to the task of wind speed estimation in satellite imagery of tropical cyclones. We provide a detailed evaluation of predictive uncertainty estimates from state-of-the-art uncertainty quantification (UQ) methods for DNNs. We find that predictive uncertainties can be utilized to further improve accuracy and analyze the predictive uncertainties of different methods across storm categories.

1 INTRODUCTION

The tremendous success of Deep Learning approaches to natural images is increasingly being explored on EO data that is becoming available in ever greater quantities (Tuia et al., 2023). Due to their often vast global coverage, EO data is an indispensable source of information for assessing the state of our planet as well as extreme events that are increasing in frequency and intensity (Kikstra et al., 2022). One category of such extreme events are tropical cyclones. Tropical cyclones - in the US alone - have lead to 6,789 deaths and caused financial damages amounting to a staggering $1,333.6 billion between 1980-2022, with an average instance cost of $22.2 billion and covering 53.9% of all costs caused by US extreme weather disasters (Smith, 2020). Although, satellite data and other in-situ measurements are often available, reliable wind speed estimation remains a challenging task. For example in October 2023, hurricane Otis underwent a rapid intensification of almost 80 kts in 12 hours before causing devastating damage in the city Acapulco.[1] The failure of satellite based wind speed estimation methods (Krämer, 2023) and the need for improving these has been highlighted after this tropical cyclone[2]. Moreover, rapidly intensifying storms near coastlines have shown a trend to become more frequent (Li et al., 2023) and, hence, this demonstrates the need for improved monitoring of wind speeds and better prediction methods to yield improved warning systems. Because data to train such prediction methods can be limited and unevenly distributed, making a perfect prediction is not always possible. However, based on the general viability of DNNs for predicting and estimating wind speeds from satellite data (see e.g. Pradhan et al. (2017)), one possible approach is to equip DNNs with modern uncertainty-quantification (UQ) methods to enhance the quality of predictions and mitigate data imbalances, as well as label and input noise. This uncertainty is important for EO applications, as in practice, a prediction model is only an element of a complex decision making process. For instance, the confidence in the prediction of a

[1] "Hurricane Otis Causes Catastrophic Damage in Acapulco, Mexico", NOAA accessed 31.01.2024.

[2] "Hurricane Otis smashed into Mexico and broke records. Why did no one see it coming?" accessed 31.01.2024.

tropical cyclone category is a key factor for deciding on public safety measures. This paper has the following contribution: Using the dataset proposed in Maskey et al. (2021), we show that equipping DNNs with predictive uncertainty can be utilized to further improve accuracy via selective prediction based on predictive uncertainty. To the best of our knowledge no previous related work (see Section 1.1) considered an evaluation of uncertainty aware regression models in this domain. We compare state-of-the-art UQ methods, (see Section 3), and demonstrate differences across storm categories according to the Saffir-Simpson scale and different dataset splits. We show that UQ can improve real-time wind speed estimation and thus outline the way to apply UQ to DNN forecasting models by a detailed assessment of existing UQ methods.

1.1 Related Work

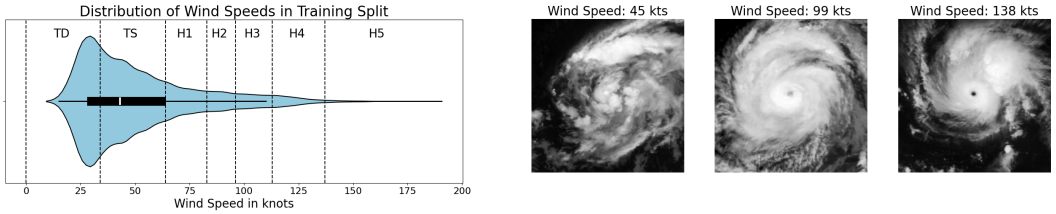
Several works have tackled the task of applying Deep Learning methods to tropical cyclone intensity estimation as a classification (Wimmers et al., 2019) or regression (Chen et al., 2019; Ma et al., 2024; Zhang et al., 2021) task. Based on a dataset of 25k images of infrared satellite imagery matched with storm data from the HURDAT2 database (Landsea & Franklin, 2013), Pradhan et al. (2017) train a CNN architecture for storm-category classification, as well es wind-speed estimation, and demonstrate improvements over previously applied statistical techniques like Advanced Dvorak Technique (ADT) (Piñeros et al., 2011), and Deviation-Angle Variance Technique (DAVT) (Ritchie et al., 2014). Maskey et al. (2020) improve the dataset quality and size by using GEOS Geostationary Operational Environmental Satellite (GEOS) and demonstrate a live production system. Our work is mostly comparable to Maskey et al. (2020) as we use their published dataset that was part of the Driven Data Challenge (Maskey et al., 2021).

2 Tropical Cyclone Dataset

Dataset name	Satellite	Spatial Res	Temporal Res	Train Samples	Val Samples	Test Samples
Tropical Cyclone	GOES	2km	15 min	53k	11k	43k

Table 1: Dataset Overview

The imagery represents single channel long-wave infrared measurements captured every 15 minutes, at 10.3 microns, that can capture the spatial structure of the storm in terms of measurements of the brightness temperature, as seen in Figure 1b. For more details about dataset collection, we refer the reader to the methodology section of (Maskey et al., 2020). We resize the images to 224x224 pixels and employ common image augmentations during training. We follow the datasplits by storm of the challenge and use dataloading available through the TorchGeo library (Stewart et al., 2022), which yields 53k training, 11k validation and 43k test samples. As Figure 1a shows, the distribution of targets is highly skewed with the majority of samples falling beneath hurricane categories defined by the Saffir Simpson Scale, Simpson (1974). We conduct experiments with the full target range but also subsets that only contain hurricane categories.



(a) Label distribution and storm categories.

(b) Dataset samples

Figure 1: Visualization of Tropical Cyclone Dataset.

3 Methods

Given a set of input-target pairs $\mathcal{D}_{\text{train}} = \{(x_i, y_i)\}_{i=1}^{N}$, (x_i, y_i), the task of the neural network is to predict a target $y^\star \in Y$ given an input $x^\star \in X$. The input is a triplet of monochrome satellite

images at time steps $[t-2, t-1, t]$ and the target is the maximum sustained wind speed in knots (kts) at time step t. [3] This is sometimes referred to as "nowcasting". For this task, we compare five classes of UQ methods: deterministic, ensemble, Bayesian, quantile and diffusion based methods. Firstly, deterministic UQ methods use a DNN, $f_\theta : X \to \mathcal{P}(Y)$, that map inputs x to the parameters of a probability distribution $f_\theta(x^\star) = p_\theta(x^\star) \in \mathcal{P}(Y)$. These include Deep Evidential Networks (**DER**) Amini et al. (2020), where we use the correction proposed by Meinert et al. (2023), and Mean Variance Networks (**MVE**) (Nix & Weigend, 1994) which output the mean and standard deviation of a Gaussian distribution $f_\theta^{\text{MVE}}(x^\star) = (\mu_\theta(x^\star), \sigma_\theta(x^\star))$. Secondly, the broadly considered state-of-the-art method Deep Ensembles (**DeepEnsembles**) proposed by Lakshminarayanan et al. (2017) utilizes an ensemble over MVE networks. Thirdly, Bayesian methods aim at modelling a distribution over the network parameters and are commonly used to approximate the first and second moment of a marginalized distribution. These include Bayesian Neural Networks with Variational Inference (**BNN VI ELBO**) Blundell et al. (2015), MC-Dropout (**MCDropout**) Gal & Ghahramani (2016), the Laplace Approximation (**Laplace**) Ritter et al. (2018)Daxberger et al. (2021) and **SWAG** Maddox et al. (2019) with partially stochastic variants presented in Sharma et al. (2023). Gaussian Process based methods model a distribution over functions that also approximate the fist and second moment of the marginalized distribution. These include Deep Kernel Learning (**DKL**) Wilson et al. (2016) and an extension thereof Deterministic Uncertainty Estimation (**DUE**) (van Amersfoort et al., 2021). Fourthly, quantile based models $f_\theta : X \to Y^n$ that map to n quantiles, $f_\theta(x^\star) = (q_1(x^\star), ..., q_n(x^\star)) \in Y^n$, such as Quantile Regression (**Quantile Regression**) and the conformalized version thereof (**ConformalQR**) suggested by Romano et al. (2019). Lastly, we also consider a diffusion model (**CARD**) as introduced by Han et al. (2022). A detailed description of the methods is provided in the supplementary material. Depending on underlying assumptions UQ, methods are regarded to express two different types of uncertainties (Hüllermeier & Waegeman, 2021). **Aleatoric uncertainty** refers to inherent randomness in the data and **epistemic uncertainty** to a lack of knowledge in the modelling process. From a statistical perspective Gruber et al. (2023) allude that such a distinction is often not possible. Thus, we focus solely on predictive uncertainty.

Evaluation methodology: In addition to standard metrics for regression, such as root-mean-squared error (RMSE), we utilize proper scoring rules such as the negative log-likelihood (NLL) and continuous ranked probability score (CRPS) (Gneiting & Raftery, 2007) and the mean absolute calibration error (MACE). To evaluate the merit of UQ methods for decision making, we use selective prediction as a downstream task. Here, samples with a predictive uncertainty above a given threshold are omitted from prediction and can be referred to an expert or other estimation methods. Based on the Saffir-Simpson Scale (Simpson, 1974) bin intervals, we chose the threshold such that it would on average shift the category of the regression prediction. Hence, we take the threshold to be the mean over categories of the wind speed interval from categories 1 to 4, which is approximately 9 kts. We experiment with different threshold choices which are reported in the supplementary material and in Fig. 3b. All methods have an ImageNet pretrained ResNet-18 (He et al., 2016) backbone available frome the timm library (Wightman, 2019). Metrics are computed with the UQ-toolbox by Chung et al. (2021).[4]

4 Results

We show fine grained results for storm categories Tropical Depression (TD), and Hurricane categories 1, 3, and 5 for better visualization. Additional results for different dataset splits and thresholds including all categories are included in the supplementary material.

How effective is selective prediction? As Table 2 shows, selective prediction - enabled through uncertainty aware models - can yield significant accuracy improvements for selected methods. The best performing methods obtain an RMSE between $9.27 - 10.95$ kts, yet the accuracy improvement obtained by selective prediction varies significantly. However, the coverage - the remaining samples after selective prediction - also varies considerably. For higher hurricane categories, accuracy and uncertainty metrics worsen substantially as shown in Figure 2 and different ranges of improvement are obtained by selective prediction, as shown in the supplementary material. When averaging

[3] We choose this input image composition, as it was utilized in the winning solution of the challenge (Maskey et al., 2021), which improved reported accuracy significantly compared to (Maskey et al., 2020).

[4] Code available under https://github.com/nilsleh/tropical_cyclone_uq

UQ group	Method	RMSE ↓	RMSE Δ ↑	Coverage ↑	CRPS ↓	NLL ↓	MACE ↓
None	Deterministic	10.50	0.00	1.00	NaN	NaN	NaN
Deterministic	**MVE**	9.95	2.10	**0.62**	**5.31**	**3.64**	0.04
	DER	10.14	NaN	0.00	10.07	4.60	0.35
Quantile	QR	10.95	3.28	0.44	5.82	3.73	**0.01**
	CQR	10.95	**6.18**	0.08	5.98	3.79	0.10
Ensemble	Deep Ensemble	16.19	0.00	0.00	8.83	4.15	0.05
Bayesian	MC Dropout	10.23	6.12	0.00	5.78	3.81	0.16
	SWAG	9.78	5.42	0.11	5.40	3.71	0.13
	Laplace	10.53	0.00	0.00	7.96	4.31	0.28
	BNN VI ELBO	**9.27**	0.00	1.00	6.28	52.60	0.41
	DKL	12.59	0.00	0.00	6.84	3.95	0.06
	DUE	9.95	0.00	0.00	5.43	3.73	0.08
Diffusion	CARD	10.86	1.50	0.60	5.84	3.92	0.05

Table 2: Evaluation Results on test set. RMSE Δ shows the improvement after selective prediction, where 0.00 indicates that all samples were withdrawn, while Coverage denotes the fraction of remaining samples that were not omitted. Threshold 9 kts.

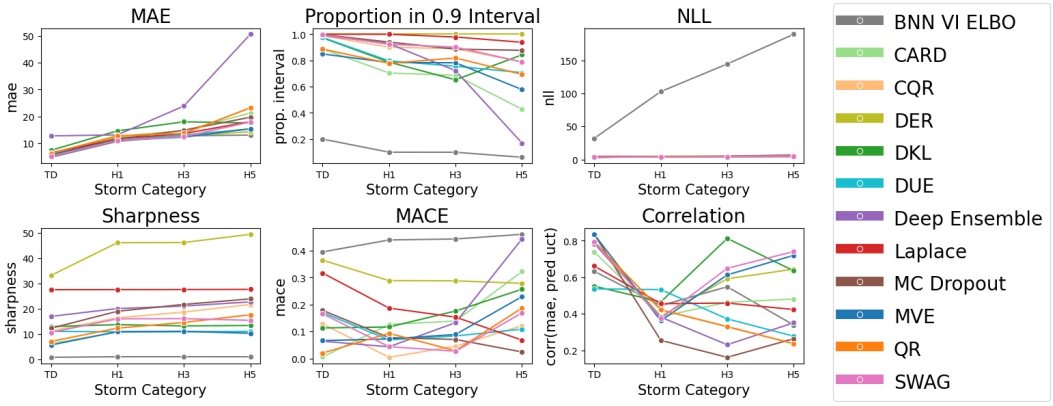


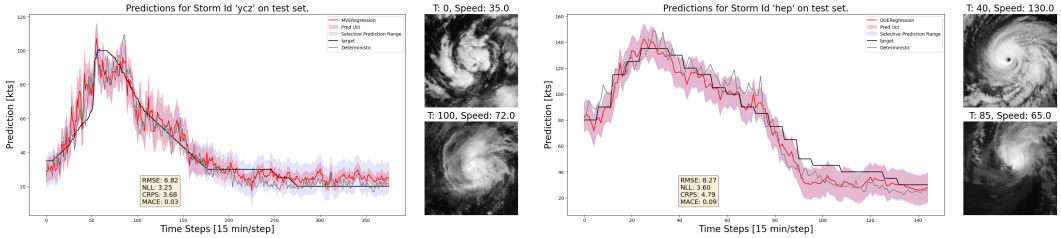
Figure 2: Uncertainty Metrics over different storm categories. We find that VI BNNs under cover (e.g. see proportion in interval), DER tends to over cover (e.g. see sharpness), with many other methods performing in between.

over all categories Table 2 shows that SWAG and CQR obtain relatively low RMSE after seletive prediction, 4.36 and 4.77 kts, while maintaining a coverage of 11% and 8%.

Error and Predictive Uncertainty across Categories: We evaluate the predictive uncertainty across storm categories with three criteria: the correlation between predictive uncertainty and MAE, sharpness, and MACE. Fig. 2, on the bottom right, shows the correlation is best for the TD case and is fairly consistent for most models. On the higher categories H3 and H5 we observe a larger spread between models, with SWAG, MVE, DKL and DER demonstrating higher correlation values. Accurate predictive uncertainties need to be both well calibrated - obtain a low MACE - and be sharp Kuleshov et al. (2018). MVE, SWAG, QR and CQR most closely fulfill this criteria. In contrast, Laplace and MC-Dropout obtain a low MACE on higher categories but are also less sharp and show lower correlation. Fig. 3a gives a "qualitative" example of MVE predictions for a selected storm track which generally follows the trend of the underlying target. Samples with a predictive uncertainty that exceeds the selective prediction threshold could be referred to an expert or postprocessing step.

4.1 Detailed discussion per UQ method group

Deterministic UQ methods: Table 2 shows MVE obtains an RMSE of 7.85 kts after selective prediction while maintaining a coverage of 62% and the lowest scoring rules, NLL and CRPS, which may be correlated to the fact that the loss objective is the NLL. At the same time MVE remains well calibrated compared to all other methods. Table 1 in the Appendix, Section 1, shows that MVE also obtains a comparably low RMSE and NLL per category. DER obtains a higher RMSE and no improvement with selective prediction, Table 2. This may be due to the fact that the predicted standard deviations of DER are relatively high compared to the selective prediction threshold, which is reflected in the sharpness accross storm categories in Figure 2. **Quantile based UQ methods**:



(a) MVE prediction Example with a visualized threshold of 9 kts.

(b) DUE Prediction Example with a visualized threshold of 12 kts.

Figure 3: Predictive Uncertainty Examples. Note that models under our setup do not have a concept of time, we merely combine individual nowcasting predictions into a time-series. Red shaded areas exceeding blue areas indicate samples that *would* be omitted during selective prediction. Figure inspired by Zhang et al. (2019).

CQR obtains higher improvements with selective prediction than QR, see Table 2, which is due to conformalization of quantiles and the resulting shift in predictive uncertainty. Yet this comes at the cost of a significantly lower coverage of CQR after selective prediction with only 8% compared to 44% for QR. **Ensemble methods**: Table 2 shows that overall Deep Ensembles obtain a higher RMSE than all other methods and also a significantly higher RMSE on category 5 cyclones, see Figure 2. Although Deep Ensembles are considered state-of-the-art, Seligmann et al. (2024) also find that they do not perform best at every UQ task. As for DER the predictive uncertainty of Deep Ensembles is larger than the selective prediction threshold, resulting in no improvement in RMSE. However, choosing a different threshold may result in accuracy improvements. We hypothesize that the variance of ensemble members might not be large enough and instead have converged to similar solutions, which implies that the ensemble members have similar biases. **Bayesian methods**: MC Dropout obtains an RMSE improvement for selective prediction, resulting in 4.11 kts at the cost of a coverage of approximately 0 %. This means that after selective predictions almost no samples remain, potentially adapting the threshold may result in improvements. SWAG obtains significant improvements with selective prediction at the cost of a low coverage of 11% and obtains relatively low CRPS and NLL as well as MACE averaged over categories, see Table 2, as well as per category, Figure 2. This indicates a good fit, however the coverage after selective prediction may be improved with a different threshold. Laplace obtains no improvement with selective prediction and interestingly also has a constant sharpness across categories as Figure 2 shows. This may be due to the fact that the Laplace approximation uses a second order Taylor expansion with respect to the model parameters of the loss and does not take into account variances in the data to construct a Gaussian approximation to the posterior weight distribution. BNN VI ELBO interestingly obtains the lowest RMSE per category and overall, Table 1 in the Appendix, Section 1, which indicates a good fit of the mean prediction. However, the predictive uncertainties are relatively small as the low sharpness and high negative log likelihood per category suggest, Figure 2. DKL obtains a relatively high RMSE and no improvement with selective prediction, although the correlation between predictive uncertainty and MAE, Figure 2, is also high on higher categories. However this may be due to high errors and high uncertainties. Compared to DKL, DUE obtains a significantly lower RMSE which may be due to the spectral normalization of layers, as this is the only difference between the methods. Otherwise DUE obtains a lower MACE per category than DKL, yet also a lower correlation between predictive uncertainty and MAE. **Diffusion UQ methods**, surprisingly CARD obtains a average RMSE and a significant improvement with selective prediction, while maintaining a coverage of 60 % and a low miscalibration error (MACE) of 0.05.

5 Conclusion

We presented a first analysis of predictive uncertainty for cyclone wind speed estimation. The various methods considered performed quite differently across storm categories and often exhibited a tradeoff between coverage vs accuracy. When predicting the maximum sustained wind speed, MVE demonstrated high coverage and low RMSE. Yet if a lower coverage is tolerable, then SWAG is a more attractive option due to it having a better RMSE than MVE. In future work, we plan to consider autoregressive models for the time series task presented in Figure 3.

6 ACKNOWLEDGEMENTS

This work was supported by the Helmholtz Association's Initiative and Networking Fund on the HAICORE@KIT partition.

REFERENCES

Alexander Amini, Wilko Schwarting, Ava Soleimany, and Daniela Rus. Deep evidential regression. *Advances in Neural Information Processing Systems*, 33:14927–14937, 2020.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International conference on machine learning*, pp. 1613–1622. PMLR, 2015.

Buo-Fu Chen, Boyo Chen, Hsuan-Tien Lin, and Russell L Elsberry. Estimating tropical cyclone intensity by satellite imagery utilizing convolutional neural networks. *Weather and Forecasting*, 34(2):447–465, 2019.

Youngseog Chung, Ian Char, Han Guo, Jeff Schneider, and Willie Neiswanger. Uncertainty toolbox: an open-source library for assessing, visualizing, and improving uncertainty quantification. *arXiv preprint arXiv:2109.10254*, 2021.

Erik Daxberger, Agustinus Kristiadi, Alexander Immer, Runa Eschenhagen, Matthias Bauer, and Philipp Hennig. Laplace redux-effortless bayesian deep learning. *Advances in Neural Information Processing Systems*, 34:20089–20103, 2021.

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

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

Cornelia Gruber, Patrick Oliver Schenk, Malte Schierholz, Frauke Kreuter, and Göran Kauermann. Sources of uncertainty in machine learning–a statisticians' view. *arXiv preprint arXiv:2305.16703*, 2023.

Xizewen Han, Huangjie Zheng, and Mingyuan Zhou. Card: Classification and regression diffusion models. *Advances in Neural Information Processing Systems*, 35:18100–18115, 2022.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016. doi: 10.1109/CVPR.2016.90.

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

Jarmo S Kikstra, Zebedee RJ Nicholls, Christopher J Smith, Jared Lewis, Robin D Lamboll, Edward Byers, Marit Sandstad, Malte Meinshausen, Matthew J Gidden, Joeri Rogelj, et al. The ipcc sixth assessment report wgiii climate assessment of mitigation pathways: from emissions to global temperatures. *Geoscientific Model Development*, 15(24):9075–9109, 2022.

Katrina Krämer. Daily briefing: Why forecasters failed to predict hurricane otis. *Nature*, 2023.

Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. Accurate uncertainties for deep learning using calibrated regression. In *International conference on machine learning*, pp. 2796–2804. PMLR, 2018.

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

Christopher W Landsea and James L Franklin. Atlantic hurricane database uncertainty and presentation of a new database format. *Monthly Weather Review*, 141(10):3576–3592, 2013.

Yi Li, Youmin Tang, Shuai Wang, Ralf Toumi, Xiangzhou Song, and Qiang Wang. Recent increases in tropical cyclone rapid intensification events in global offshore regions. *Nature Communications*, 14(1):5167, 2023.

Zhaoyang Ma, Yunfeng Yan, Jianmin Lin, and Dongfang Ma. A multi-scale and multi-layer feature extraction network with dual attention for tropical cyclone intensity estimation. *IEEE Transactions on Geoscience and Remote Sensing*, 2024.

Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. *Advances in neural information processing systems*, 32, 2019.

M. Maskey, R. Ramachandran, I. Gurung, B. Freitag, M. Ramasubramanian, and J. Miller. Tropical Cyclone Wind Estimation Competition Dataset. https://doi.org/10.34911/rdnt.xs53up, 2021.

Manil Maskey, Rahul Ramachandran, Muthukumaran Ramasubramanian, Iksha Gurung, Brian Freitag, Aaron Kaulfus, Drew Bollinger, Daniel J Cecil, and Jeffrey Miller. Deepti: Deep-learning-based tropical cyclone intensity estimation system. *IEEE journal of selected topics in applied Earth observations and remote sensing*, 13:4271–4281, 2020.

Nis Meinert, Jakob Gawlikowski, and Alexander Lavin. The unreasonable effectiveness of deep evidential regression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 9134–9142, 2023.

David A Nix and Andreas S Weigend. Estimating the mean and variance of the target probability distribution. In *Proceedings of 1994 ieee international conference on neural networks (ICNN'94)*, volume 1, pp. 55–60. IEEE, 1994.

Miguel F Piñeros, Elizabeth A Ritchie, and J Scott Tyo. Estimating tropical cyclone intensity from infrared image data. *Weather and forecasting*, 26(5):690–698, 2011.

Ritesh Pradhan, Ramazan S Aygun, Manil Maskey, Rahul Ramachandran, and Daniel J Cecil. Tropical cyclone intensity estimation using a deep convolutional neural network. *IEEE Transactions on Image Processing*, 27(2):692–702, 2017.

Elizabeth A Ritchie, Kimberly M Wood, Oscar G Rodríguez-Herrera, Miguel F Piñeros, and J Scott Tyo. Satellite-derived tropical cyclone intensity in the north pacific ocean using the deviation-angle variance technique. *Weather and forecasting*, 29(3):505–516, 2014.

Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable laplace approximation for neural networks. In *6th International Conference on Learning Representations, ICLR 2018-Conference Track Proceedings*, volume 6. International Conference on Representation Learning, 2018.

Yaniv Romano, Evan Patterson, and Emmanuel Candes. Conformalized quantile regression. *Advances in neural information processing systems*, 32, 2019.

Florian Seligmann, Philipp Becker, Michael Volpp, and Gerhard Neumann. Beyond deep ensembles: A large-scale evaluation of bayesian deep learning under distribution shift. *Advances in Neural Information Processing Systems*, 36, 2024.

Mrinank Sharma, Sebastian Farquhar, Eric Nalisnick, and Tom Rainforth. Do bayesian neural networks need to be fully stochastic? In *International Conference on Artificial Intelligence and Statistics*, pp. 7694–7722. PMLR, 2023.

Robert H Simpson. The hurricane disaster—potential scale. *Weatherwise*, 27(4):169–186, 1974.

Adam B. Smith. U.s. billion-dollar weather and climate disasters, 1980 - present (ncei accession 0209268), 2020. URL https://www.ncei.noaa.gov/archive/accession/0209268.

Adam J Stewart, Caleb Robinson, Isaac A Corley, Anthony Ortiz, Juan M Lavista Ferres, and Arindam Banerjee. Torchgeo: deep learning with geospatial data. In *Proceedings of the 30th international conference on advances in geographic information systems*, pp. 1–12, 2022.

Devis Tuia, Konrad Schindler, Begüm Demir, Gustau Camps-Valls, Xiao Xiang Zhu, Mrinalini Kochupillai, Sašo Džeroski, Jan N van Rijn, Holger H Hoos, Fabio Del Frate, et al. Artificial intelligence to advance earth observation: a perspective. *arXiv preprint arXiv:2305.08413*, 2023.

Joost van Amersfoort, Lewis Smith, Andrew Jesson, Oscar Key, and Yarin Gal. On feature collapse and deep kernel learning for single forward pass uncertainty. *arXiv preprint arXiv:2102.11409*, 2021.

Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019.

Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P Xing. Deep kernel learning. In *Artificial intelligence and statistics*, pp. 370–378. PMLR, 2016.

Anthony Wimmers, Christopher Velden, and Joshua H Cossuth. Using deep learning to estimate tropical cyclone intensity from satellite passive microwave imagery. *Monthly Weather Review*, 147(6):2261–2282, 2019.

Chang-Jiang Zhang, Xiao-Jie Wang, Lei-Ming Ma, and Xiao-Qin Lu. Tropical cyclone intensity classification and estimation using infrared satellite images with deep learning. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 14:2070–2086, 2021.

Rui Zhang, Qingshan Liu, and Renlong Hang. Tropical cyclone intensity estimation using two-branch convolutional neural network from infrared and water vapor images. *IEEE Transactions on Geoscience and Remote Sensing*, 58(1):586–597, 2019.

UNCERTAINTY AWARE TROPICAL CYCLONE WIND SPEED ESTIMATION FROM SATELLITE DATA
- APPENDIX -

Nils Lehmann
Technical University of Munich
n.lehmann@tum.de

Nina Maria Gottschling
MF-DAS OP - EO Data Science, DLR
nina-maria.gottschling@dlr.de

Stefan Depeweg
Siemens AG
stefan.depeweg@siemens.com

Eric Nalisnick
University of Amsterdam
e.t.nalisnick@uva.nl

1 ADDITIONAL FIGURES AND TABLES

1.1 EXPERIMENTS WITH A MINIMUM WIND SPEED OF ZERO

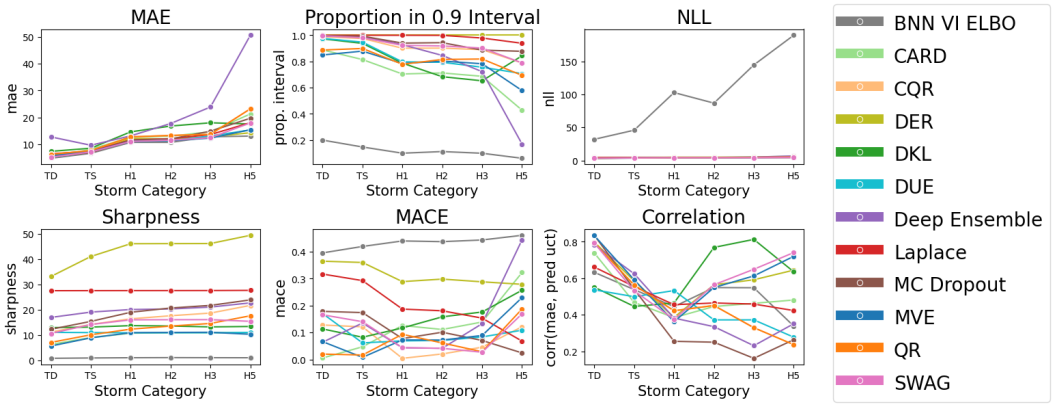


Figure 1: Uncertainty Metrics over different storm categories.

	uqmethod	TD rmse ↓	TD nll ↓	TS rmse ↓	TS nll ↓	H1 rmse ↓	H1 nll ↓	H2 rmse ↓	H2 nll ↓	H3 rmse ↓	H3 nll ↓	H4 rmse ↓	H4 nll ↓	H5 rmse ↓	H5 nll ↓
Deterministic	MVE	7.504	**3.264**	9.313	3.692	13.801	4.330	14.504	4.219	**15.911**	4.337	**14.843**	**4.149**	18.175	4.691
	DER	7.581	4.432	9.185	4.651	14.902	4.802	15.178	4.800	16.553	4.809	14.946	4.812	18.644	4.885
Quantile	CQR	8.319	3.529	9.723	3.801	15.508	4.209	16.635	4.280	17.972	4.430	17.401	4.345	27.543	4.934
	QR	8.319	3.359	9.723	3.731	15.508	4.370	16.635	4.422	17.972	4.690	17.401	4.404	27.543	5.286
Ensemble	Deep Ensemble	15.562	4.038	12.150	4.028	15.901	4.297	21.370	4.582	28.335	5.038	36.286	5.432	52.112	6.817
Bayesian	**BNN VI ELBO**	**6.479**	31.978	**8.361**	45.774	**13.405**	102.664	**13.377**	86.707	17.175	144.454	15.703	105.009	**16.581**	189.350
	BNN VI	8.152	3.422	8.918	**3.617**	14.409	4.355	17.683	4.613	22.457	5.288	25.110	5.004	33.325	5.224
	Laplace	8.401	4.281	9.343	4.293	14.308	4.371	15.299	4.390	17.773	4.444	16.493	4.416	22.843	4.580
	MC Dropout	7.374	3.565	8.643	3.808	14.401	4.204	15.263	4.256	19.763	4.501	19.454	4.475	24.063	4.624
	DKL	9.634	3.757	11.552	3.892	17.243	4.367	19.994	4.542	22.006	4.776	15.287	4.172	19.217	4.535
	DUE	7.036	3.523	9.139	3.662	13.948	4.126	14.375	4.177	17.870	4.651	16.312	4.427	21.046	5.177
	SWAG	6.940	3.427	9.051	3.775	13.805	**4.103**	14.761	**4.115**	16.307	**4.187**	16.083	4.155	20.497	**4.495**
Diffusion	CARD	7.167	3.268	10.158	4.045	15.557	4.816	16.622	4.858	19.077	5.329	17.497	4.785	24.322	6.378

Table 1: Evaluation Results on test set. RMSE per category.

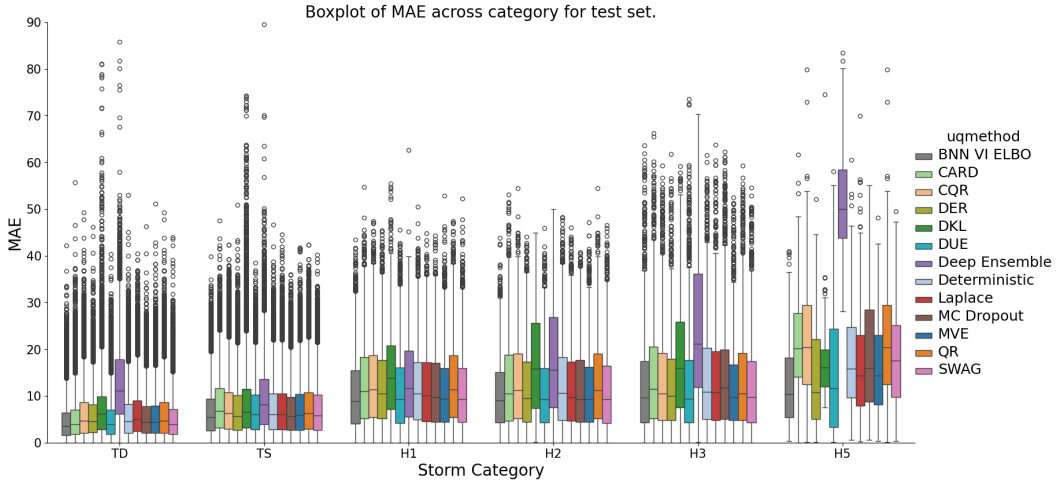


Figure 2: MAE over different storm categories.

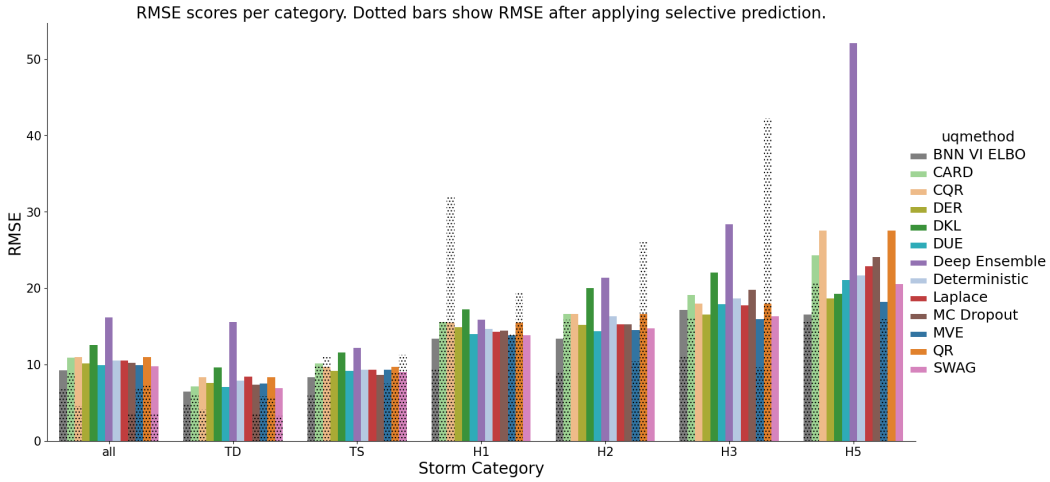


Figure 3: RMSE before and after selective prediction over different storm categories. The dotted bars are the RMSE after selective prediction with a threshold of 9 kts.

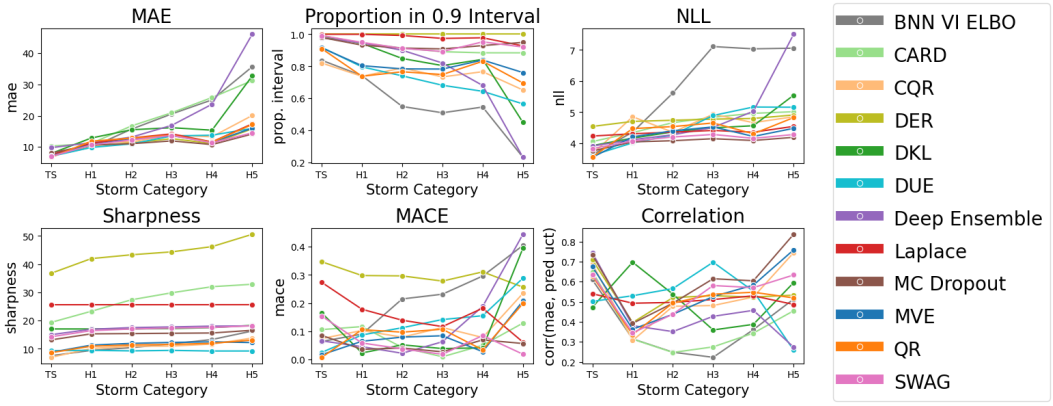


Figure 4: Uncertainty Metrics over different storm categories.

1.2 Experiments with a minimum wind speed of 34

UQ group	Method	RMSE ↓	RMSE Δ ↑	Coverage ↑	CRPS ↓	NLL ↓	MACE ↓
None	Deterministic	11.68	0.00	**1.00**	NaN	NaN	NaN
Deterministic	MVE	11.21	NaN	NaN	6.08	**3.79**	0.03
	DER	**11.05**	NaN	0.00	10.24	**4.60**	0.33
Quantile	QR	11.33	2.26	0.44	6.11	3.86	0.03
	CQR	11.57	1.99	0.62	6.25	4.00	0.08
Ensemble	Deep Ensemble	14.56	NaN	0.00	7.96	4.06	0.04
Bayesian	MC Dropout	11.56	2.55	0.01	6.40	3.85	0.07
	SWAG	**11.09**	NaN	NaN	6.26	3.93	0.12
	Laplace	11.68	NaN	0.00	7.98	4.27	0.24
	BNN VI ELBO	13.10	1.73	0.68	7.11	4.20	0.10
	DKL	13.17	NaN	NaN	7.43	4.05	0.11
	DUE	**11.07**	NaN	0.00	**6.03**	3.85	**0.02**
Diffusion	CARD	15.09	**4.60**	**0.02**	8.83	4.22	0.09

Table 2: Evaluation Results on test set. RMSE Δ shows the improvement after selective prediction, while Coverage denotes the fraction of remaining samples that were not omitted. Threshold 9 knots.

1.3 EXPERIMENTS WITH A MINIMUM WIND SPEED OF 64

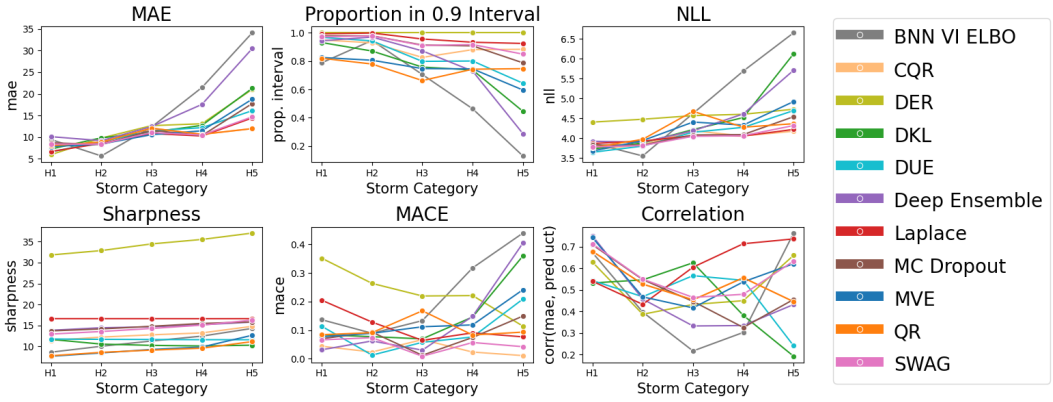


Figure 5: Uncertainty Metrics over different storm categories.

UQ group	Method	RMSE ↓	RMSE Δ ↑	Coverage ↑	CRPS ↓	NLL ↓	MACE ↓
None	Deterministic	10.78	0.00	**1.00**	NaN	NaN	NaN
Deterministic	MVE	11.48	1.43	0.64	6.45	3.95	0.09
	DER	11.90	NaN	0.00	9.33	4.47	0.29
Quantile	QR	11.76	1.20	0.62	6.68	4.03	0.10
	CQR	11.76	**2.46**	0.05	6.55	3.87	0.02
Ensemble	Deep Ensemble	14.43	NaN	0.00	8.08	4.07	**0.01**
Bayesian	MC Dropout	11.86	NaN	0.00	6.70	3.91	0.06
	SWAG	11.59	NaN	0.00	6.50	3.87	0.05
	Laplace	**10.74**	NaN	0.00	6.37	3.94	0.15
	BNN VI ELBO	13.82	2.44	0.44	7.87	4.19	0.11
	DKL	12.12	NaN	0.00	6.80	3.98	0.02
	DUE	11.38	NaN	0.00	**6.31**	**3.85**	0.03

Table 3: Evaluation Results on test set. RMSE Δ shows the improvement after selective prediction, while Coverage denotes the fraction of remaining samples that were not omitted. Threshold 9 knots.

2 Overview of applied UQ Methods

We consider regression problems in the following setting: Given an input $x^\star \in X$ the task is to predict a target $y^\star \in Y$. Notably, regression is distinguishable from classification as the targets are continuous and possibly infinite, as opposed to a fixed set of finite labels in classification. In our experiments we want to use a neural network to predict a unobserved test target $y^\star \in Y$ for a given unobserved test input $x^\star \in X$. Precisely, given the set of $n \in \mathbb{N}$ observed training input-target pairs from our dataset,

$$\mathcal{D}_{\text{train}} = \{(x_i, y_i)\}_{i=1}^n, \tag{1}$$

the task of the models or NNs is to predict a target $y^\star \in Y$ given an input $x^\star \in X$ such that the loss objective between the predictions and targets is minimized over all the training points. This is described in the following.

The model or NN can be regarded as a function f_θ, parameterized by weights θ, that maps inputs x directly to targets $y \in Y$,

$$f_\theta : X \to Y \tag{2}$$

or to a probability distribution,

$$f_\theta : X \to \mathcal{P}(Y) \tag{3}$$

such that

$$f_\theta(x^\star) = p_\theta(x^\star) \in \mathcal{P}(Y). \tag{4}$$

or as $f_\theta : X \to Y^n$ that maps to n quantiles,

$$f_\theta(x^\star) = (q_1(x^\star), ..., q_n(x^\star)) \in Y^n. \tag{5}$$

For example, a NN can be configured to output the mean and standard deviation of a Gaussian distribution $f_\theta(x^\star) = (\mu_\theta(x^\star), \sigma_\theta(x^\star))$.

Previous Benchmarks and Reviews of Uncertainty Quantification Methods for Regression Problems

Publication	(Gustafsson et al., 2023)	(Schmähling et al., 2022)	(Dewolf et al., 2022)	(Izmailov et al., 2021)	here
Deterministic Methods					
Baseline					✓
Gaussian (MVE)	✓				✓
Deep Evidential Networks (DER)					✓
Ensemble based					
Deep Ensembles, GMM	✓	✓	✓		✓
Bayesian					
MC Dropout, GMM		✓	✓		✓
BNN with VI				✓	✓
Laplace Approximation					✓
SWAG				✓	✓
DVI, SI				✓	
HMC				✓	
Gaussian Process based					
"Gaussian Process (GP)"			✓		
Approximate GP			✓		
Deep Kernel Learning (DKL)					✓
Spectral Normalized GPs (DUE)					✓
Quantile based					
Quantile Regression (QR)	✓		✓		✓
Conformal Prediction (CQR)	✓		✓		✓
Diffusion Model					
CARD					✓

Table 4: Comparison of previous reviews. The BNN implementations of BNN with VI and SWAG in this work use partially stochastic networks, as proposed in (Sharma et al., 2023).

As Table 4 demonstrates, we compare five classes of UQ methods: deterministic, ensemble, Bayesian, quantile and diffusion based methods. Firstly, deterministic UQ methods use a DNN, $f_\theta : X \to \mathcal{P}(Y)$, that map inputs x to the parameters of a probability distribution $f_\theta(x^\star) = p_\theta(x^\star) \in \mathcal{P}(Y)$. These include Deep Evidential Networks (**DER**) Amini et al. (2020), where we use the correction proposed by Meinert et al. (2023), and Mean Variance Networks (**MVE**) (Nix & Weigend, 1994) which output the mean and standard deviation of a Gaussian distribution $f_\theta^{\text{MVE}}(x^\star) = (\mu_\theta(x^\star), \sigma_\theta(x^\star))$. Secondly, the broadly considered state-of-the-art method Deep Ensembles (**DeepEnsembles**) proposed by Lakshminarayanan et al. (2017) utilizes an ensemble over

MVE networks. Thirdly, Bayesian methods aim at modelling a distribution over the network parameters and are commonly used to approximate the first and second moment of a marginalized distribution. These include Bayesian Neural Networks with Variational Inference (**BNN VI ELBO**) Blundell et al. (2015), MC-Dropout (**MCDropout**) Gal & Ghahramani (2016), the Laplace Approximation (**Laplace**) Ritter et al. (2018)Daxberger et al. (2021a) and **SWAG** Maddox et al. (2019). A slightly different approach is taken by Gaussian process based methods that model a distribution over functions, yet also approximate the fist and second moment of this distribution. These include Deep Kernel Learning (**DKL**) Wilson et al. (2016) and an extension thereof Deterministic Uncertainty Estimation (**DUE**) (van Amersfoort et al., 2021). Fourthly, quantile based models $f_\theta : X \to Y^n$ that map to n quantiles, $f_\theta(x^\star) = (q_1(x^\star), ..., q_n(x^\star)) \in Y^n$, such as Quantile Regression (**Quantile Regression**) and the conformalized version thereof (**ConformalQR**) suggested by Romano et al. (2019). Lastly, we also consider a diffusion model (**CARD**) as introduced by Han et al. (2022). A detailed description of the methods is provided in the supplementary material.

3 Description of UQ Methods

Baseline model: Depending on the application a DNN, for example a residual network, that is used as a baseline. This model does not predict any uncertainty and just a mean $f_\theta(x^\star)$. For the loss objective, the mean squared error is used

$$\mathcal{L}(\theta, (x^\star, y^\star)) = (f_\theta(x^\star) - y^\star)^2. \tag{6}$$

3.1 Deterministic UQ Methods

In the following we list the deterministic UQ methods considered in this work. These methods provide UQ estimates within a single forward pass by predicting the parameters of a probability distribution.

Gaussian: The Gaussian model, also referred to as Mean Variance Estimation, first studied in Nix & Weigend (1994) and further used in Sluijterman et al. (2023), is a deterministic model that predicts the parameters of a Gaussian distribution

$$f_\theta(x^\star) = (\mu_\theta(x^\star), \sigma_\theta(x^\star)) \tag{7}$$

in a single forward pass, where standard deviations $\sigma_\theta(x^\star)$ can be used as a measure of data uncertainty. To this end, the network now outputs two parameters and is trained with the Gaussian negative log-likelihood (NLL) as a loss objective Kendall & Gal (2017), that is given by

$$\mathcal{L}(\theta, (x^\star, y^\star)) = \frac{1}{2}\ln\left(2\pi\sigma_\theta(x^\star)^2\right) + \frac{1}{2\sigma_\theta(x^\star)^2}\left(\mu_\theta(x^\star) - y^\star\right)^2. \tag{8}$$

Correspondingly, the model prediction consists of a predictive mean, $\mu_\theta(x^\star)$, and the predictive uncertainty, in this case the standard deviation $\sigma_\theta(x^\star)$.

Deep Evidential Networks (DER):

Deep Evidential Regression (DER) (Amini et al., 2020) is a single forward pass UQ method that aims to disentangle aleatoric and epistemic uncertainty. DER entails a four headed network output

$$f_\theta(x^\star) = (\gamma_\theta(x^\star), \nu_\theta(x^\star), \alpha_\theta(x^\star), \beta_\theta(x^\star)), \tag{9}$$

that is used to compute the predictive t-distribution with $2\alpha(x^\star)$ degrees of freedom:

$$p(y(x^\star)|f_\theta(x^\star)) = \mathrm{St}_{2\alpha_\theta(x^\star)}\left(y^\star \middle| \gamma_\theta(x^\star), \frac{\beta_\theta(x^\star)(1+\nu_\theta(x^\star))}{\nu_\theta(x^\star)\alpha_\theta(x^\star)}\right). \tag{10}$$

In Amini et al. (2020) the network weights are obtained by minimizing the loss objective that is the negative log-likelihood of the predictive distribution and a regularization term. However, due to several drawbacks of DER, Meinert et al. (2023) propose the following adapted loss objective that we also utilise,

$$\mathcal{L}(\theta, (x^\star, y^\star)) = \log \sigma_\theta^2(x^\star) + (1 + \lambda \nu_\theta(x^\star)) \frac{(y^\star - \gamma_\theta(x^\star))^2}{\sigma_\theta^2(x^\star)} \tag{11}$$

where $\sigma_\theta^2(x^\star) = \beta_\theta(x^\star)/\nu_\theta(x^\star)$. The mean prediction is given as,

$$\mu_\theta(x^\star) = \gamma_\theta(x^\star). \tag{12}$$

Further following (Meinert et al., 2023), we use their reformulation of the uncertainty decomposition. The aleatoric uncertainty is given by

$$u_{\text{aleatoric}}(x^\star) = \sqrt{\frac{\beta(x^\star)}{\alpha(x^\star) - 1}}, \tag{13}$$

and the epistemic uncertainy by,

$$u_{\text{epistemic}}(x^\star) = \frac{1}{\sqrt{\nu(x^\star)}}. \tag{14}$$

The predictive uncertainty is then, given by

$$u(x^\star) = \sqrt{u_{\text{epistemic}}(x^\star)^2 + u_{\text{aleatoric}}(x^\star)^2}. \tag{15}$$

3.2 Ensemble Based UQ Methods

Deep Ensembles: introduced in Lakshminarayanan et al. (2017), Deep Ensembles approximate a posterior distribution over the model weights with a Gaussian mixture model over the output of separately initialized and trained networks. In Wilson & Izmailov (2020) the authors showed that Deep Ensembles can be interpreted as a Bayesian method.

For the Deep Ensembles model the predictive mean is given by the mean taken over $N \in \mathbb{N}$ models $f_{\theta_i}(x^\star) = \mu_{\theta_i}(x^\star)$ that output a mean with different weights $\{\theta_i\}_{i=1}^N$,

$$\mu(x^\star) = \frac{1}{N} \sum_{i=1}^{N} \mu_{\theta_i}(x^\star). \tag{16}$$

The predictive uncertainty is given by the standard deviation of the predictions of the N different networks, Gaussian ensemble members,

$$\sigma(x^\star) = \sqrt{\frac{1}{N} \sum_{i=1}^{N} \left(\mu_{\theta_i}(x^\star) - \mu(x^\star)\right)^2}. \tag{17}$$

Summary of hyperparameters for the Deep Ensembles model

Hyperparameter	value range	hints
Number of ensemble members	$N \approx [5, 20]$	do an ablation study on N.

Deep Ensembles GMM:

For the Deep Ensembles GMM model, the predictive mean is given by the mean taken over $N \in \mathbb{N}$ models $f_{\theta_i}(x^\star) = (\mu_{\theta_i}(x^\star), \sigma_{\theta_i}(x^\star))$ with different weights $\{\theta_i\}_{i=1}^N$,

$$\mu_g(x^\star) = \frac{1}{N}\sum_{i=1}^{N} \mu_{\theta_i}(x^\star). \tag{18}$$

The predictive uncertainty is given by the standard deviation of the Gaussian mixture model consisting of the N different networks, Gaussian ensemble members,

$$\sigma_g(x^\star) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\mu_{\theta_i}(x^\star) - \mu_g(x^\star)\right)^2 + \frac{1}{N}\sum_{i=1}^{N}\sigma_{\theta_i}^2(x^\star)}. \tag{19}$$

Note that the difference between "Deep Ensembles" and "Deep Ensembles GMM" is that in the latter we also consider the predictive uncertainty output of each individual ensemble member, whereas in the former we only consider the means and the variance of the mean predictions of the ensemble members.

Because each ensemble member has a probabilistic predictive distribution $(\mu_{\theta_i}(x^\star), \sigma_{\theta_i}(x^\star))$, we can also perform a decomposition into epistemic and aleatoric components:

$$u_{\text{epistemic}}(x^\star) = \frac{1}{N}\sum_{i=1}^{N}\left(\mu_g(x^\star) - \mu_{\theta_i}(x^\star)\right)^2 , \tag{20}$$

$$u_{\text{aleatoric}}(x^\star) = \frac{1}{N}\sum_{i=1}^{N}\sigma_{\theta_i}^2(x^\star) . \tag{21}$$

Summary of hyperparameters for the Deep Ensembles model

Hyperparameter	value range	hints
Number of ensemble members	$N \approx [5, 20]$	do an ablation study on N.

3.3 Bayesian UQ Methods

The general aim of Bayesian UQ methods is to obtain the predictive distribution by marginalization over the model weights θ,

$$p(y^\star|x^\star, D) = \int p(y^\star|x^\star, \theta)p(\theta|D)d\theta. \tag{22}$$

The posterior distribution over the weights $p(\theta|D)$ can be approximated by utilizing Bayes' theorem or, for example, by a variational approach. However, the predictive distribution, equation 22, is usually intractable and, in the following various approaches of approximation are presented (most of which rely on sampling over the posterior).

MC-Dropout: Is an approximate Bayesian method with sampling. A fixed dropout rate $p \in [0, 1)$ is used, meaning that random weights are set to zero during each forward pass with the probability p. This models the network weights and biases as a Bernoulli distribution with dropout probability p. While commonly used as a regularization method, Gal & Ghahramani (2016) showed that activating dropout during inference over multiple forward passes yields an approximation to the posterior over the network weights. Due to its simplicity it is widely adopted in practical applications, but MC-Dropout and variants thereof have also been criticized for their theoretical shortcomings Hron et al. (2017), Osband (2016).

For the MC Dropout model the prediction consists of a predictive mean and a predictive uncertainty. For the predictive mean, the mean is taken over $m \in \mathbb{N}$ forward passes through the network $f_{p,\theta}$ with a fixed dropout rate p, resulting in different weights $\{\theta_i\}_{i=1}^m$, given by

$$f_p(x^\star) = \frac{1}{m}\sum_{i=1}^{m} f_{p,\theta_i}(x^\star). \tag{23}$$

The predictive uncertainty is given by the standard deviation of the predictions over m forward passes,

$$\sigma_p(x^\star) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(f_{p,\theta_i}(x^\star) - f_p(x^\star)\right)^2}. \tag{24}$$

Summary of hyperparameters for the MC Dropout model

Hyperparameter	value range	hints
Drop out rate	$p \in [0, 1)$	start with $p = 0.2$.

MC Dropout GMM: We also consider combining this method with the previous model Gaussian network, as in Kendall & Gal (2017), aiming at disentangling the data and model uncertainties, abbreviated as MC Dropout GMM. For the MC Dropout GMM model, the prediction again consists of a predictive mean and a predictive uncertainty $f_{p,\theta}(x^\star) = (\mu_{p,\theta}(x^\star), \sigma_{p,\theta}(x^\star))$. Here the predictive mean is given by the mean taken over m forward passes through the Gaussian network mean predictions $\mu_{p,\theta}$ with a fixed dropout rate p, resulting in different weights $\{\theta_i\}_{i=1}^m$, given by

$$\mu_p(x^\star) = \frac{1}{m}\sum_{i=1}^{m} \mu_{p,\theta_i}(x^\star). \tag{25}$$

The predictive uncertainty is given by the standard deviation of the Gaussian mixture model obtained by the predictions over m forward passes,

$$\sigma_p(x^\star) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(\mu_{p,\theta_i}(x^\star) - \mu_p(x^\star)\right)^2 + \frac{1}{m}\sum_{i=1}^{m}\sigma^2_{p,\theta_i}(x^\star)}. \tag{26}$$

A decomposition of uncertainty can then be performend in a similar way as to with deep ensembles.

Summary of hyperparameters for the MC Dropout GMM model

Hyperparameter	value range	hints
Number of burn-in-epochs	$\approx [0, n]$	after burn-in-epochs train variance and mean outputs.
Drop out rate	$p \in [0, 1)$	start with $p = 0.2$.

BNN with VI: Bayesian Neural Networks (BNNs) with variational inference (VI) are an approximate Bayesian method. Here, we follow the mean-field assumption, meaning that the variational distribution is factorized as a product of individual Gaussian distributions. This results in a diagonal Gaussian approximation of the posterior distribution over the model parameters

The most common approach is to maximize the evidence lower bound (ELBO). We note that there are other, alternative approaches for variational inference, such as α-divergence minimization (Hernandez-Lobato et al., 2016).

Uitilizing standard stochastic gradient descent by using the reparameterization trick Kingma & Welling (2013) one can backpropagate through the necessary sampling procedure, a process called

Monte Carlo variational Bayes (Ranganath et al., 2014).

The predictive likelihood is given by,

$$p(Y|\theta, X) = \prod_{i=1}^{N} p(y_i|\theta, x_i) = \prod_{i=1}^{N} \mathcal{N}(y_i|f_\theta(x_i), \Sigma). \tag{27}$$

The prior on the weights is given by,

$$p(\theta) = \prod_{l=1}^{L} \prod_{h=1}^{V_l} \prod_{j=1}^{V_{l-1}+1} \mathcal{N}(w_{hj,l}|0, \lambda) \tag{28}$$

where $w_{hj,l}$ is the h-th row and the j-th column of weight matrix θ_L at layer index L and λ is the prior variance. Note that as we use partially stochastic networks, equation 28 may contain less factors $\mathcal{N}(w_{hj,l}|0, \lambda)$ depending on how many layers are stochastic. Then, the posterior distribution of the weights is obtained by Bayes' rule as

$$p(\theta|\mathcal{D}) = \frac{p(Y|\theta, X)p(\theta)}{p(Y|X)}. \tag{29}$$

As the posterior distribution over the weights is intractable a variational approximation is used,

$$q(\theta) \approx p(\theta|\mathcal{D}), \tag{30}$$

that is a diagonal Gaussian. Now given an input $x^\star$, the predictive distribution can be obtained as

$$p(y^\star|x^\star, \mathcal{D}) = \int p(y^\star|\theta, x^\star)p(\theta|\mathcal{D})d\theta. \tag{31}$$

As equation 31 is intractable it is approximated by sampling form the approximation $q(\theta)$ in equation 30 to the posterior distribution in equation 29. The parameters of $q(\theta)$ are obtained by maximizing the evidence lower bound (ELBO) on the Kullback-Leibler (KL) divergence between the variational approximation and the posterior distribution over the weights. The negative ELBO is given by,

$$\mathcal{L}(\theta, (x^{star}, y^{star})) = \beta D_{KL}(q(\theta)||p(\theta)) + \frac{1}{2}\ln\left(2\pi\sigma^2\right) + \frac{1}{2\sigma^2}\left(f_\theta(x^\star) - y^\star\right)^2. \tag{32}$$

The KL divergence can be computed analytically in the case of a Gaussian prior and the hyperparameter β can be used to weight the influence of the variational parameters relative to that of the data. Alternatively, in the case of a fixed dataset of size N this parameter is automatically set to $\frac{1}{N}$. The hyperparameter σ can be either fixed or set to be an additional parameter to be tuned by including it in the objective function Eq. (32), a process called type-II maximum likelihood.

The predictive mean is obtained as the mean of the network output f_θ with S weight samples from the variational approximation $\theta_s \sim q(\theta)$,

$$f_m(x^\star) = \frac{1}{S}\sum_{i=1}^{S} f_{\theta_s}(x^\star). \tag{33}$$

The predictive uncertainty is given by the standard deviation thereof, including the (possibly estimated) constant output noise σ:

$$\sigma_p(x^\star) = \sqrt{\frac{1}{S}\sum_{i=1}^{S}\left(f_{\theta_s}(x^\star) - f_m(x^\star)\right)^2 + \sigma^2}. \tag{34}$$

If one uses the NLL and adapts the BNN to output a mean and standard deviation of a Gaussian $f_{\theta_s}(x^\star) = (\mu_{\theta_s}(x^\star), \sigma_{\theta_s}(x^\star))$, the mean prediction is given by

$$f_m(x^\star) = \frac{1}{S}\sum_{s=1}^{S} \mu_{\theta_s}(x^\star). \tag{35}$$

and the predictive uncertainty is obtained as the standard deviation of the corresponding Gaussian mixture model obtained by the weight samples,

$$\sigma_p(x^\star) = \sqrt{\frac{1}{S}\sum_{s=1}^{S}\left(\mu_{\theta_s}(x^\star) - f_m(x^\star)\right)^2 + \sum_{s=1}^{S}\sigma^2_{\theta_s}(x^\star)}. \tag{36}$$

Summary of hyperparameters for the BNN with VI model

Hyperparameter	value range	hints
Number burn-in-epochs	$\approx [0, n]$	after burn-in-epochs train variance and mean outputs.
Loss scale factor β	$\beta \approx [100, 500]$	should depend on parameter and train set size.
Samples during training S_{tr}	$S_{tr} \approx [5, 20]$	depending on network size and computing resources.
Samples during tests and prediction S_{te}	$S_{te} \approx [5, 50]$	depending on network size and computing resources.
Output noise scale σ	$\sigma \approx [1.0, 5.0]$	depending on label noise.
Prior mean μ_p for stochastic parameters	$\mu_p \approx [0, 1.0]$	start with 0. Prior variance σ_p for stochastic parameters
$\sigma_p \approx [0, 3.0]$	start with 1.0.	
Mean initialization for posterior μ_{pr}	$\mu_{pr} \approx [0, 1.0]$	approximate posterior over parameters
Variance initialization for posterior ρ_{pr}	$\rho_{pr} \approx [-6.0, 0.0]$	variance through $\sigma = \log(1 + \exp(\rho))$, approximate posterior over parameters
Bayesian layer type	"flipout" or "reparameterization"	
Stochastic module names	list of module names or a list of module numbers	Transform module to be stochastic.

Laplace Approximation: Originally introduced by MacKay (1992), the Laplace Approximation has been adapted to modern neural networks by Ritter et al. (2018) and Daxberger et al. (2021a) and is an approximate Bayesian method. The goal of the Laplace Approximation is to use a second-order Taylor expansion around the fitted MAP estimate and yield a posterior approximation over the model parameters via a full-rank, diagonal or Kronecker-factorized approach. In order for the Laplace Approximation to be computationally feasible for larger network architectures, we use the Laplace library to include approaches, such as subnetwork selection that have been for example proposed by Daxberger et al. (2021b).

The general idea of the Laplace Approximation to obtain a distribution over the network parameters with a Gaussian distribution centered at the MAP estimate of the parameters Daxberger et al. (2021b). In this setting, a prior distribution $p(\theta)$ is defined over our network parameters. Because modern neural networks consists of millions of parameters, obtaining a posterior distribution over the weights θ is intractable. The LA takes MAP estimate of the parameters θ_{MAP} from a trained network $f_{\theta_{MAP}}(x) = \mu_{\theta_{MAP}}(x)$ and constructs a Gaussian distribution around it. The parameters θ_{MAP} are obtained by

$$\theta_{MAP} = \text{argmin}\mathcal{L}(\theta; D), \tag{37}$$

where $\mathcal{L}$ is the mean squared error or also referred to as the ℓ^2 loss, $\mathcal{L}(\theta; \mathcal{D}) := -\sum_{i=1}^{n} log(p(y_i|f_\theta(x_i)))$ and the posterior $p(y_i|f_\theta(x_i))$ is chosen to be a Gaussian with constant variance σ^2, such that the loss is the mean squared error and a homoskedastic noise model is assumed. Then with Bayes Theorem, as in Daxberger et al. (2021b), one can relate the posterior to the loss,

$$p(\theta|D) = p(D|\theta)p(\theta)/p(D) = \frac{1}{Z}exp(-\mathcal{L}(\theta; D)), \tag{38}$$

with $Z = \int p(D|\theta)p(\theta)d\theta$. Now a second-order expansion of $\mathcal{L}$ around θ_{MAP} is used to construct a Gaussian approximation to the posterior $p(\theta|D)$:

$$-\mathcal{L}(\theta; D) \approx -\mathcal{L}(\theta_{MAP}; D) - \frac{1}{2}(\theta - \theta_{MAP})(\nabla^2_\theta \mathcal{L}(\theta; D)|\theta_{MAP})(\theta - \theta_{MAP}). \tag{39}$$

The term with the first order derivative is zero as the loss is evaluated at a minimum θ_{MAP} Murphy (2022), and, further, one assumes that the first term is neglible as the loss is evaluated at $\theta = \theta_{MAP}$. Then taking the expontential of both sides allows to identify, after normalization, the Laplace approximation,

$$p(\theta|D) \approx \mathcal{N}(\theta_{MAP}, \Sigma) \qquad \text{with} \qquad \Sigma = (\nabla^2_\theta \mathcal{L}(\theta; D)|\theta_{MAP})^{-1}. \tag{40}$$

As the covariance is just the inverse Hessian of the loss, with $\theta_{MAP} \in \mathcal{R}^W$ and $H^{-1} \in \mathcal{R}^{W \times W}$, with W being the number of weights, the posterior distribution is given by

$$p(\theta|D) \approx \mathcal{N}(\theta_{MAP}, H^{-1}). \tag{41}$$

The computation of the Hessian term is still expensive. Therefore, further approximations are introduced in practice, most commonly the Generalized Gauss-Newton matrix Martens (2020). This takes the following form:

$$H \approx \widetilde{H} = \sum_{n=1}^{N} J_n^T H_n J_n, \tag{42}$$

where $J_n \in \mathcal{R}^{O \times W}$ is the Jacobian of the model outputs with respect to the parameters θ and $H_n \in \mathcal{R}^{O \times O}$ is the Hessian of the negative log-likelihood with respect to the model outputs, where O denotes the model output size and W the number of parameters.

Given equation 41 during inference on unseen data, one cannot compute the full posterior predictive distribution but instead resort to sampling $\theta_s \sim p(\theta|D)$ for $s \in \{1, ..., S\}$ to approximate the predictions,

$$\hat{y}(x^\star) = \frac{1}{S} \sum_{s=1}^{S} f_{\theta_s}(x^\star), \tag{43}$$

and obtain the predictive uncertainty by

$$\sigma^2(x^\star) = \sqrt{\frac{1}{S} \sum_{s=1}^{S} f_{\theta_s}(x^\star)^2 - \hat{y}(x^\star)^2 + \sigma^2}. \tag{44}$$

For the subnet strategy, we include the options from the Laplace library for selecting the stochastic parameters.

Summary of hyperparameters for the BNN with VI model

Hyperparameter	value range	hints
Number burn-in-epochs	$\approx [0, n]$	after burn-in-epochs train variance and mean outputs.
Loss scale factor β	$\beta \approx [100, 500]$	should depend on parameter and train set size.
Samples during training S_{tr}	$S_{tr} \approx [5, 20]$	depending on network size and computing resources.
Samples during tests and prediction S_{te}	$S_{te} \approx [5, 50]$	depending on network size and computing resources.

SWAG: Is an approximate Bayesian method and uses a low-rank Gaussian distribution as an approximation to the posterior over model parameters. The quality of approximation to the posterior over model parameters is based on using a high SGD learning rate that periodically stores weight parameters in the last few epochs of training Maddox et al. (2019). SWAG is based on Stochastic Weight Averaging (SWA), as proposed in Izmailov et al. (2018). For SWA the weights are obtained by minimising the MSE loss with a variant of stochastic gradient descent. After, a number of burn-in epochs, $\tilde{t} = T - m$, the last m weights are stored and averaged to obtain an approximation to the posterior, by

$$\theta_{SWA} = \frac{1}{m} \sum_{t=\tilde{t}}^{T} \theta_t. \tag{45}$$

For SWAG we use the implementation as proposed by Maddox et al. (2019). Here the posterior is approximated by a Gaussian distribution with the SWA mean, equation 45 and a covariance matrix over the stochastic parameters that consists of a low rank matrix plus a diagonal,

$$p(\theta|D)\mathcal{N}\left(\theta_{SWA}, \frac{1}{2}(\Sigma_{diag} + \Sigma_{low-rank})\right). \tag{46}$$

The diagonal part of the covariance is given by

$$\Sigma_{diag} = \text{diag}(\bar{\theta^2} - \theta_{SWA}^2) \tag{47}$$

where,

$$\bar{\theta^2} = \frac{1}{m} \sum_{t=\tilde{t}}^{T} \theta_t^2. \tag{48}$$

The low rank part of the covariance is given by

$$\Sigma_{low-rank} = \frac{1}{m} \sum_{t=\tilde{t}}^{T} (\theta_t - \bar{\theta}_t)(\theta_t - \bar{\theta}_t)^T, \tag{49}$$

where $\bar{\theta}_t$ is the running estimate of the mean of the parameters from the first t epochs or also samples. In order to approximate the mean prediction, we again resort to sampling from the posterior equation 46. With $\theta_s \sim p(\theta|D)$ for $s \in \{1, ..., S\}$, the mean prediction is given by

$$\hat{y}(x^\star) = \frac{1}{S} \sum_{s=1}^{S} f_{\theta_s}(x^\star), \tag{50}$$

and obtain the predictive uncertainty by

$$\sigma(x^\star) = \sqrt{\frac{1}{S} \sum_{s=1}^{S} f_{\theta_s}(x^\star)^2 - \hat{y}(x^\star)^2}. \tag{51}$$

For the subnet strategy, we include selecting the parameters to be stochastic by module names.

3.4 Gaussian Process Based UQ Methods

Recap of Gaussian Processes (GPs): The goal of previously introduced methods was to find a distribution over the weights of a parameterized function i.e. a neural network. In contrast, the basic idea of a Gaussian Process (GP) is to instead consider a distribution over possible functions, that fit the data in some way. Formally,

"A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution." Seeger (2004)

Precisely, a GP can be described by a possibly infinite amount of function values

$$f(x) \sim \mathcal{GP}(m(x), k_\gamma(x)), \tag{52}$$

such that any finite collection of function values f has a joint Gaussian distribution,

$$f = f(X) = [f(x_1), \ldots, f(x_K)]^\top \sim \mathcal{N}(m_X, \mathcal{K}_{X,X}), \tag{53}$$

with a mean vector, $(m_X)i = m(x_i)$, and covariance matrix, $(\mathcal{K}_{X,X})_{ij} = k_\gamma(x_i, x_j)$, stemming from the mean function m and covariance kernel of the GP, k_γ, that is parametrized by γ. A commonly used covariance function is the squared exponential, also referred to as Radial Basis Function (RBF) kernel, exponentiated quadratic or Gaussian kernel:

$$k_\gamma(x, x') = \text{cov}(f(x), f(x')) = \eta^2 \exp\left(-\frac{1}{2l^2}|x - x'|^2\right). \tag{54}$$

Where $\gamma = (\eta^2, l)$ and η^2 can be set to 1 or tuned as a hyperparameter. By default the lengthscale $l = 1$ but can also be optimized over. Now the GP, $f(x) \sim GP(m(x), k(x, x'))$, as a distribution over functions can be used to solve a regression problem. Following Seeger (2004), consider the simple case where the observations are noise free and you have training data $\mathcal{D}_{\text{train}} = \{(x_i, y_i)\}_{i=1}^N$ with $X = (x_i)_{i=1}^N$ and $Y = (y_i)_{i=1}^N$. The joint prior distribution of the training outputs, Y, and the test outputs $f_* = f_*(X_*) = (f(i_k))_{i=1}^m$ where $X_* = (x_i)_{i=1}^m$ are the test points, according to the prior is

$$p(Y, f_*) = \mathcal{N}\left(0, \begin{bmatrix} \mathcal{K}_{X,X} & \mathcal{K}_{X,X_*} \\ \mathcal{K}_{X_*,X} & \mathcal{K}_{X_*,X_*} \end{bmatrix}\right). \tag{55}$$

Here the mean function is assumed to be $m_X = 0$ and $\mathcal{K}_{X,X_*}$ denotes the $N \times m$ matrix of the covariances evaluated at all pairs of training and test points, and similarly for the other entries $\mathcal{K}_{X,X}$, $\mathcal{K}_{X_*,X_*}$ and $\mathcal{K}_{X_*,X}$. To make predictions based on the knowledge of the training points, conditioning on the prior observations is used and yields,

$$\begin{aligned} p(f_*|X_*, X, Y) &= \mathcal{N}(\mathcal{K}_{X_*,X}\mathcal{K}_{X,X}^{-1}Y, \mathcal{K}_{X_*,X_*} - \mathcal{K}_{X_*,X_*}\mathcal{K}_{X,X}^{-1}\mathcal{K}_{X,X_*}) \\ &= \mathcal{N}(m(X, X_*, Y), \tilde{\mathcal{K}}_{X,X_*}). \end{aligned}$$

Now to generate function values on test points, one uses samples from the posterior distribution $f_*(X_*) \sim \mathcal{N}(m(X, X_*, Y), \tilde{K}(X, X_*))$. To illustrate how we can obtain these samples from the posterior distribution, consider a Gaussian with arbitrary mean m and covariance K, i.e. $f_* \sim \mathcal{N}(m, K)$. For this one can use a scalar Gaussian generator, which is available in many packages:

1. Compute the Cholesky decomposition of $K = LL^T$, where L is a lower triangular matrix. This works because K is symmetric by definition.
2. Then, draw multiple $u \sim \mathcal{N}(0, I)$.
3. Now, compute the samples with $f_* = m + Lu$. This has the desired mean, m and covariance $L\mathbb{E}(uu^T)L^T = LL^T = K$.

The above can be extended to incorporate noisy measurements $y \to y + e$, see Seeger (2004), or noise on the inputs as in Johnson et al. (2019). Both of these extensions require tuning of further hyperparameters, yet beneficially allow to incorporate a prediction of aleatoric uncertainty in a GP.

For example, assume additive Gaussian noise on the distribution of the function values,

$$p(y(x)|f(x)) = \mathcal{N}(y(x); f(x), \sigma^2). \tag{56}$$

Then the predictive distribution of the GP evaluated at the K_* test points, X_*, is given by

$$\begin{aligned} p(f_*|X_*, X, Y, \gamma, \sigma^2) &= \mathcal{N}(\mathbb{E}[f_*], \text{cov}(f_*)) \,, \\ \mathbb{E}[f_*] &= m_{X_*} + \mathcal{K}_{X_*,X}[\mathcal{K}_{X,X} + \sigma^2 I]^{-1} Y \,, \\ \text{cov}(f_*) &= \mathcal{K}_{X_*,X_*} - \mathcal{K}_{X_*,X}[\mathcal{K}_{X,X} + \sigma^2 I]^{-1} \mathcal{K}_{X,X_*} \,. \end{aligned} \tag{57}$$

Here m_{X_*} is the $K_* \times 1$ mean vector, which is assumed to be zero in the previous case.

In both cases, with and without additive noise on the function values, the GP is trained by learning interpretable kernel hyperparameters. The log marginal likelihood of the targets y - the probability of the data conditioned only on kernel hyperparameters γ - provides a principled probabilistic framework for kernel learning:

$$\log p(y|\gamma, X) \propto -\left(y^\top (\mathcal{K}_\gamma + \sigma^2 I)^{-1} y + \log |\mathcal{K}_\gamma + \sigma^2 I|\right) \,, \tag{58}$$

where $\mathcal{K}_\gamma$ is used for $\mathcal{K}_{X,X}$ given γ. Kernel learning can be achieved by optimizing Eq. equation 58 with respect to γ.

The computational bottleneck for inference is solving the linear system $(\mathcal{K}_{X,X} + \sigma^2 I)^{-1} y$, and for kernel learning it is computing the log determinant $\log |\mathcal{K}_{X,X} + \sigma^2 I|$ in the marginal likelihood. The standard approach is to compute the Cholesky decomposition of the $K \times K$ matrix $\mathcal{K}_{X,X}$, which requires $\mathcal{O}(K^3)$ operations and $\mathcal{O}(K^2)$ storage. After inference is complete, the predictive mean costs $\mathcal{O}(K)$, and the predictive variance costs $\mathcal{O}(K^2)$, per test point x_*.

Deep Kernel Learning (DKL): Conceptually DKL consists of a NN architecture that extracts a feature representation of the input x and fits an approximate GP on top of these features to produce a probabilistic output Wilson et al. (2016). DKL combines GPs and DNNs in a scalable way. In practice, all parameters, the weights of the feature extractor and the GP parameters are optimized jointly by maximizing the log marginal likelihood of the GP. We utilize GPytorch for our implementation Gardner et al. (2018) and use a grid approximation where we optimized over the number of inducing points. For DKL the GP is transformed by replacing the inputs x by the outputs of a NN in the following way. The kernel $k_\gamma(x, x')$ with hyperparameters θ is replaced by,

$$k_\gamma(x, x') \to k_\gamma(g(x, \theta), g(x', \theta)) \,, \tag{59}$$

where $g(x, \theta)$ is a non-linear mapping given by a deep architecture, such as a deep convolutional network mapping into a feature space of dimension J, parametrized by weights θ,

$$\begin{aligned} g(\cdot, \theta) : X &\to \mathbb{R}^J \\ x &\mapsto g(x, \theta). \end{aligned} \tag{60}$$

This so called deep kernel in equation 59 is now used as the covariance function of a GP to model data $\mathcal{D} = \{x_i, y_i\}_{i=1}^N$. The deep kernel hyperparameters, $\rho = \{\gamma, \theta, \sigma^2\}$, can be *jointly* learned by maximizing the log *marginal likelihood* of the GP equation 61.

$$\mathcal{L} = \log p(Y|\gamma, X, \theta) \propto -\left(y^\top (K_{\gamma,\theta} + \sigma^2 I)^{-1} y + \log |K_{\gamma,\theta} + \sigma^2 I|\right) \,, \tag{61}$$

Except for the replacement of input data, one can almost follow the same procedures for learning and inference as for GPs as outlined previously. For optimizing equation 61 the chain rule is used to compute derivatives of the log marginal likelihood with respect to the deep kernel hyperparameters as in Wilson et al. (2016).

Exact inference is possible for the regression case, yet the computational complexity scales cubically with the number of data points and makes it not suitable for large datasets. Thus, following (van Amersfoort et al., 2021) in the implementation the sparse GP of (Titsias, 2009) and the variational approximation of (Hensman et al., 2014) is used, in order to allow for DKL to scale to large training

datasets. The sparse GP approximation of (Titsias, 2009) augments the DKL model with M inducing inputs, $Z \in \mathbb{R}^{M \times J}$, where J is the dimensionality of the feature space, as in equation 60. Moreover, to perform computationally efficient inference we use the the variational approximation introduced by (Hensman et al., 2014), where inducing points Z are treated as variational parameters. U are random variables with prior

$$p(U) = \mathcal{N}(U|m_Z, \mathcal{K}_{Z,Z}), \tag{62}$$

and variational posterior

$$q(U) = \mathcal{N}(U|\widetilde{m}, S), \tag{63}$$

where $\widetilde{m} \in \mathbb{R}^M$ and $S \in \mathbb{R}^{M \times M}$ are variational parameters and initialized at the zero vector and the identity matrix respectively. The approximate predictive posterior distribution at training points X is then

$$p(f|Y)q(f) = \int p(f|U)q(U)dU \tag{64}$$

Here $p(f|U)$ is a Gaussian distribution for which we can find an analytic expression, see (Hensman et al., 2014) for details. Note that we deviate from (Hensman et al., 2014) in that our input points X are mapped into feature space just before computing the base kernel, while inducing points are used as is (they are defined in feature space). The variational parameters Z, $\widetilde{m}$, and S and the feature extractor parameters θ and GP model hyparparameters γ, given by l and η^2, and σ^2 are all learned at once by maximizing a lower bound on the log marginal likelihood of the predictive distribution $p(Y|X)$, the ELBO, denoted by $\mathcal{L}$. For the variational approximation above, this is defined as

$$\log(p(Y|X)) \geq \mathcal{L}(Z, m, S, \gamma, \theta, \sigma^2) = \sum_{i=1}^{N} \mathbb{E}_{q(f)} \left[\log p(y_i|f(x_i))\right] - \beta \mathrm{D}_{\mathrm{KL}}(q(U)||p(U)). \tag{65}$$

Both terms can be computed analytically when the likelihood is Gaussian and all parameters can be learned using stochastic gradient descent. To accelerate optimization gpytorch additionally utilizes the whitening procedure of (Matthews, 2017) in their Variational Strategy. The approximate predictive posterior distribution at test points X^* is then

$$p(f_*|Y)q(f_*) = \int p(f_*|U)q(U)dU \tag{66}$$

For regression tasks we directly use the function values f_* above as the predictions. We use the mean of $p(f_*|Y)$ as the prediction, and the variance as the uncertainty.

4 Deterministic Uncertainty Estimation (DUE) - extension of DKL

DUE builds on DKL by using the same model except for exchanging the feature extractor of the DKL model. With this replacement DUE addresses limitations of DKL and provides potentially robust uncertainty estimates. According to (van Amersfoort et al., 2021) with DKL, data points dissimilar to the training data (also called OOD data) can potentially be mapped close to feature representations of in-distribution points. These feature representations, which are close in some norm, input into the approximate GP yield similar or nearly the same predictions. This is called "feature collapse", and suggests that a constraint must be placed on the deep feature extractor. Based on deterministic uncertainty quantification (DUQ) (Van Amersfoort et al., 2020) and spectrally normaplized GPs (SNGP) (Liu et al., 2020), the authors of (van Amersfoort et al., 2021) propose to use a bi-Lipschitz constraint on a feature extractor. This bi-Lipschitz constraint is enforced by spectral normalization on the weights, (Miyato et al., 2018; Gouk et al., 2021). This constraint mitigates so-called "feature collapse", by forcing the feature representation to be sensitive to changes in the input (lower Lipschitz, avoids feature collapse) but also generalize due to smoothness (upper Lipschitz).

For convolutional and linear layers following (van Amersfoort et al., 2021), we use spectral normalization of the weight matrices to promote approximate bi-Lipschitz continuity. To promote spectral

Algorithm 1 Algorithm for training DUE (van Amersfoort et al., 2021)

Definitions:
- Residual NN $g_\theta : x \to \mathbb{R}^J$ with feature space dimensionality J and parameters θ.
- Approximate GP with parameters $\rho = \{\gamma, \sigma^2, \omega\}$, where $\gamma = \{l, \eta\}$ and l length scale and η output scale of k_γ, ω GP variational parameters (including m inducing point locations Z)
- Learning rate ζ, loss function $\mathcal{L}$

Using a random subset of p points of our training data, $X^{\text{init}} \subset X$, compute:
Initial inducing points: K-means on $g_\theta(X^{\text{init}})$ with $K = m$. Use found centroids as initial inducing point locations Z in GP.
Initial length scale:
$l = \frac{1}{\binom{p}{2}} \sum_{i=0}^{p} \sum_{j=i+1}^{p} |g_\theta(X_i^{\text{init}}) - g_\theta(X_j^{\text{init}})|_2$.

for minibatch $x_b, y_b \subset X, Y$ **do**
  $\theta' \leftarrow$ spectral_normalization(θ)
  $p(y_b'|x_b) \leftarrow$ evaluate_GP$_\theta(g_{\theta'}(x_b))$
  $\mathcal{L} \leftarrow \text{ELBO}_\theta(p(y_b'|x_b), y_b)$
  $(\rho, \theta) \leftarrow (\rho, \theta) + \zeta * \nabla_{\rho,\theta}\mathcal{L}$
end for

normalization for fully connected layers and 1×1 convolutions online power iteration are used and for larger convolutions an approximate method, as proposed in (Gouk et al., 2021) and was first implemented by (Behrmann et al., 2019), is used. Spectral normalization is also extended to batch normalization by rescaling the weights, see (van Amersfoort et al., 2021) for details. Adding spectral normalization to batch normalization layers makes it more likely that the entire network's upper Lipschitz constant is bounded. The mean prediction and predictive uncertainty are obtained as for DKL.

Summary of learnable parameters:

- weights of DNN feature extractor θ
- for the GP, parameters γ: noise hyperparamter σ^2, the GP function mean m, the length scale of the GP kernel l and the scale of the kernel η^2. In the above case the GP hyperparameters are learned by optimizing ELBO.

Summary of hyperparameters:

- number of power iterations for spectral normalization, usually set to $r = 1$
- number of initial inducing points M

4.1 Quantile Based UQ Methods

Quantile Regression (QR): The goal of Quantile Regression is to extend a standard regression model to also predict conditional quantiles that approximate the true quantiles of the data at hand. It does not make assumptions about the distribution of errors as is usually common. It is a more commonly used method in Econometrics and Time-series forecasting Koenker & Bassett Jr (1978).

In the following we will describe univariate quantile regression. Any chosen conditional quantile $\alpha \in [0, 1]$ can be defined as

$$q_\alpha(x) := \inf\{y \in \mathbb{R} : F(y|X = x) \geq \alpha\}, \tag{67}$$

where $F(y|X = x) = P(Y \leq y|X = x)$ is a strictly monotonic increasing cumulative density function.

For Quantile Regression, the NN f_θ parameterized by θ, is configured to output the number of quantiles that we want to predict. This means that, if we want to predict p quantiles $[\alpha_1, ...\alpha_n]$,

$$f_\theta(x_\star) = (\hat{y}_1(x^\star), ..., \hat{y}_n(x^\star)). \tag{68}$$

The model is trained by minimizing the pinball loss function Koenker & Bassett Jr (1978), given by the following loss objective,

$$\mathcal{L}_i(\theta, (x^\star, y^\star)) = \max\{(1-\alpha_i)(y^\star - \hat{y}_i(x^\star)), \alpha(y^\star - \hat{y}_i(x^\star))\}. \tag{69}$$

Here $i \in \{1, ..., n\}$ denotes the number of the quantile and $100\alpha_i$ is the percentage of the quantile for $\alpha_i \in [0, 1)$. Note that for $\alpha = 1/2$ one recovers the ℓ^1 loss. During inference, the model will output an estimate for the chosen quantiles and these can be used as an indication of aleatoric uncertainty.

Conformalized Quantile Regression (CQR): Conformal Prediction is a post-hoc uncertainty quantification method to yield calibrated predictive uncertainty bands with proven coverage guarantees Angelopoulos & Bates (2021). Based on a held out calibration set, CQR uses a score function to find a desired coverage quantile $\hat{q}$ and conformalizes the QR output by adjusting the quantile bands via $\hat{q}$ for an unseen test point as follows $x_\star$:

$$T(x_\star) = [\hat{y}_{\alpha/2}(x_\star) - \hat{q}, \hat{y}_{1-\alpha/2}(x_\star) + \hat{q}] \tag{70}$$

where $\hat{y}_{\alpha/2}(x_\star)$ is the lower quantile output and $\hat{y}_{1-\alpha/2}(x_\star)$, Romano et al. (2019).

4.2 Diffusion Based UQ Methods

CARD: The classification and regression diffusion (CARD) models, as introduced in Han et al. (2022), combine a denoising diffusion-based conditional generative model and a pre-trained conditional mean estimator in order to obtain a predictive distribution given an input. Given a target $y^\star$ and input $x^\star$ CARD utilizes a series of intermediate predictions $y_{1:T}$ for a number of steps $T \in \mathbb{N}$. The parameters of the diffusion-based conditional generative model are obtained by optimising the following objective

$$\mathcal{L}_{\text{ELBO}}(y^\star, x^\star) = \mathcal{L}_0(y^\star, x^\star) + \sum_{t=2}^{T} \mathcal{L}_{t-1}(y^\star, x^\star) + \mathcal{L}_T(y^\star, x^\star), \tag{71}$$

where the individual terms are given by

$$\mathcal{L}_0(y^\star, x^\star) = \mathbb{E}_q\left[-\log(p_\theta(y^\star|y_1, x)\right] \tag{72}$$

$$\mathcal{L}_{t-1}(y^\star, x^\star) = \mathbb{E}_q\left[D_{\text{KL}}(q(y_{t-1}|y_t, y_0, x)||p_\theta(y_{t-1}|y_t, x))\right] \tag{73}$$

$$\mathcal{L}_T(y^\star, x^\star) = \mathbb{E}_q\left[D_{\text{KL}}(q(y_T|, y_0, x)||p(y_T|x))\right] \tag{74}$$

and the predictive distribution $p(y_T|x)$ is obtained by a MAP estimate, in our case the deterministic base model,

$$p(y_T|x) = \mathcal{N}(f_{\theta_{MAP}}(x), \mathbb{I}). \tag{75}$$

Following Pandey et al. (2022) the forward process of conditional distributions with a diffision schedule $(\beta_t)_{t=1}^T \in (0, 1)^T$ is defined such that a closed-form solution exists,

$$p(y_t|y_{t-1}, f_{\theta_{MAP}}(x)) = \mathcal{N}(y_t; \sqrt{1-\beta_t}y_t + (1-\sqrt{1-\beta_t})f_{\theta_{MAP}}(x), \beta_t\mathbb{I}), \tag{76}$$

this admits a closed form and non-iterative solution at each time step $t \in \{1, ..., T\}$,

$$p(y_t|y_0, f_{\theta_{MAP}}(x)) = \mathcal{N}(y_t; \sqrt{\alpha_t}y_t + (1 - \sqrt{\alpha_t})f_{\theta_{MAP}}(x), \beta_t\mathbb{I}), \tag{77}$$

with $\alpha_t = \Pi_{l=1}^{t}(1 - \beta_l)$. For regression the goal is to reverse the above diffusion process to recover the distribution of the noise term and, hence, obtaining the aleatoric uncertainty of the second moment predictive distribution $p(y|x)$. For this a neural network ϵ_θ is trained that given a sample y_t predicts the corresponding noise $\epsilon\epsilon_\theta(x, y_t, f_{\theta_{MAP}}(x), t)$. The predictive mean and uncertainty, in terms of standard deviation, is obtained by moment matching with the predictive samples y_0 approximating the labels $y^\star$.

4.3 Partially Stochastic Network Strategies

In order to adapt the Bayesian UQ methods to large EO data sets, we support partially stochastic NNs following the approach presented in Sharma et al. (2023). In Sharma et al. (2023) the authors demonstrate experimentally and theoretically that partially stochastic networks can also approximate predictive distributions. There are multiple ways to obtain partially stochastic networks. For the Laplace Approximation and SWAG methods, we use a two-stage training. First, all parameters are obtained by a MAP estimate. Then, in the second training stage the stochastic parameters are obtained. For BNN with VI and BNN+LV we use joint training, where the stochastic and deterministic parameters are learnt jointly by maximising the evidence lower bound or the so called α-divergence, Depeweg et al. (2018).

5 Metrics

Regression tasks are commonly evaluated by accuracy metrics such as Root Mean Squared Error (RMSE) or coefficient of determination, R^2. A better quality of prediction is indicated by a lower RMSE and MAE and a R^2 score close to 1.0. However, these measures only characterize the error between point predictions and available targets. When considering UQ methods, we therefore need additional metrics in the form of proper scoring rules Gneiting & Raftery (2007) which do not ignore predictive uncertainty. In particular, we consider the negative log-likelihood (NLL) of a Gaussian as a proper scoring rule, Gneiting & Raftery (2007). Moreover, we consider calibration as introduced in Kuleshov et al. (2018). As neither the NLL or calibration are sufficient to verify a useful forecast since a model with large predictive uncertainties can be well calibrated and obtain a sufficient NLL, we additionally consider sharpness, which measures the mean of the predictive uncertainties. We use Chung et al. (2021) for metric computation and some plots.

The RMSE is computed between the targets $\boldsymbol{y} = (y_i)_{i=1}^N$ and the mean model predictions $\boldsymbol{f}(x) = (f(x_i))_{i=1}^N$ for N samples as

$$\text{RMSE}(\boldsymbol{f}(x), \boldsymbol{y}) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(f(x_i) - y_i)^2}. \tag{78}$$

The MAE is computed as

$$\text{MAE}(\boldsymbol{f}(x), \boldsymbol{y}) = \frac{1}{N}\sum_{i=1}^{N}|f(x_i) - y_i|. \tag{79}$$

The R^2 is computed as

$$\text{R}^2 = \text{R}^2(\boldsymbol{f}(x), \boldsymbol{y}) = 1 - \frac{\sum_{i=1}^{N}(f(x_i) - y_i)^2}{\sum_{i=1}^{N}\left(f(x_i) - \frac{1}{N}\sum_{j=1}^{N}f(x_j)\right)^2}. \tag{80}$$

However, these measures only characterize the error between point predictions and available targets. In order to compare the predictive uncertainties to the target distribution, we need additional metrics,

such as proper scoring rules Gneiting & Raftery (2007). We consider the NLL of a Gaussian as a proper scoring rule Gneiting & Raftery (2007). We also report the miscalibration area, where a lower miscalibration area indicates a better fit of the predictive uncertainties to the true target distribution. To quantify the overall confidence of a model in a single metric, we consider sharpness which computes the mean of the predictive uncertainties. We use Chung et al. (2021) for computing these metrics.

The NLL is computed between the targets $\boldsymbol{y} = (y_i)_{i=1}^N$ and the mean model predictions $\boldsymbol{f}(x) = (f(x_i))_{i=1}^N$ and predictive uncertainties $\boldsymbol{\sigma}(x) = (\sigma(x_i))_{i=1}^N$ for N samples as NLL is computed as

$$\text{NLL}((\boldsymbol{f}(x), \boldsymbol{\sigma}(x)), \boldsymbol{y}) = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{1}{2} \ln \left(2\pi\sigma(x_i)^2 \right) + \frac{1}{2\sigma(x_i)^2} \left(f(x_i) - y_i \right)^2 \right), \tag{81}$$

Additional we consider the scoring rule of the Continuous Ranked Probability Score (CRPS), which for single sample and a predictive distribution that is Gaussian is given by

$$crps(\mathcal{N}(\mu, \sigma), y) = -\sigma \left(\frac{y-\mu}{\sigma} (2\Phi \left(\frac{y-\mu}{\sigma} \right) - 1) + 2\phi \left(\frac{y-\mu}{\sigma} \right) - \frac{1}{\sqrt{\pi}} \right), \tag{82}$$

where Φ is the cumulative density function and ϕ probability distribution of $\mathcal{N}(0, 1)$. Then, we compute the average sum over all predictions and labels, where $f_\theta(x_i^\star) = (\mu(x_i^\star), \sigma(x_i^\star))$, which gives the reported CRPS,

$$CRPS = \frac{1}{N^\star} \sum_{i=1}^{N^\star} crps(f_\theta(x_i^\star), y_i^\star). \tag{83}$$

The miscalibration area is computed based on Chung et al. (2021) and is identical to mean absolute calibration error, however the integration here is taken by tracing the area between curves.

The sharpness is computed as

$$\text{sharpness}(\boldsymbol{\sigma}(x)) = \sqrt{\frac{1}{N} \sum_{i=1}^{N} \sigma(x_i)^2}. \tag{84}$$

Another key aspect for assessing the reliability of uncertainty estimates is calibration. Calibration refers to the degree to which a predicted distribution matches the true underlying distribution of the data. The mean absolute calibration error, (MACE), gives the mean absolute error of the expected and observed proportions for a given range of quantiles.

6 Training Details

We train all methods with the SGD or Adam optimizer Kingma & Ba (2014) with default parameters for a minimum of 50 epochs until convergence based on the validation loss and evaluate the trained model on the in and out of distribution sets. BNNs, DKL, DUE and CARD were trained with Adam while other methods where trained with SGD. All methods and experiments are implemented in Pytorch Paszke et al. (2019) and Lighting Falcon & Team (2019) to enhance reproducibility of results.
For the Deep Ensemble we use five independently trained Gaussian Networks, for MC-Dropout and SWAG we use the settings suggested by Maddox et al. (2019), for the Laplace Approximation we use a Kronecker factored Hessian approximation through the Laplace library Daxberger et al. (2021a), for the DKL implementation we follow van Amersfoort et al. (2021), for the BNN we use Krishnan et al. (2022) with their default parameters and for Quantile Regression we predict quantiles 0.1, 0.5, and 0.9 Angelopoulos & Bates (2021).

REFERENCES

Alexander Amini, Wilko Schwarting, Ava Soleimany, and Daniela Rus. Deep evidential regression. *Advances in Neural Information Processing Systems*, 33:14927–14937, 2020.

Anastasios N Angelopoulos and Stephen Bates. A gentle introduction to conformal prediction and distribution-free uncertainty quantification. *arXiv preprint arXiv:2107.07511*, 2021.

Jens Behrmann, Will Grathwohl, Ricky TQ Chen, David Duvenaud, and Jörn-Henrik Jacobsen. Invertible residual networks. In *International Conference on Machine Learning*, pp. 573–582. PMLR, 2019.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International conference on machine learning*, pp. 1613–1622. PMLR, 2015.

Youngseog Chung, Ian Char, Han Guo, Jeff Schneider, and Willie Neiswanger. Uncertainty toolbox: an open-source library for assessing, visualizing, and improving uncertainty quantification. *arXiv preprint arXiv:2109.10254*, 2021.

Erik Daxberger, Agustinus Kristiadi, Alexander Immer, Runa Eschenhagen, Matthias Bauer, and Philipp Hennig. Laplace redux-effortless bayesian deep learning. *Advances in Neural Information Processing Systems*, 34:20089–20103, 2021a.

Erik Daxberger, Eric Nalisnick, James U Allingham, Javier Antorán, and José Miguel Hernández-Lobato. Bayesian deep learning via subnetwork inference. In *International Conference on Machine Learning*, pp. 2510–2521. PMLR, 2021b.

Stefan Depeweg, Jose-Miguel Hernandez-Lobato, Finale Doshi-Velez, and Steffen Udluft. Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning. In *International Conference on Machine Learning*, pp. 1184–1193. PMLR, 2018.

Nicolas Dewolf, Bernard De Baets, and Willem Waegeman. Valid prediction intervals for regression problems. *Artificial Intelligence Review*, pp. 1–37, 2022.

W Falcon and TPL Team. Pytorch lightning the lightweight pytorch wrapper for high-performance ai research. scale your models, not the boilerplate, 2019.

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

Jacob R Gardner, Geoff Pleiss, David Bindel, Kilian Q Weinberger, and Andrew Gordon Wilson. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. In *Advances in Neural Information Processing Systems*, 2018.

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

Henry Gouk, Eibe Frank, Bernhard Pfahringer, and Michael J Cree. Regularisation of neural networks by enforcing lipschitz continuity. *Machine Learning*, 110:393–416, 2021.

Fredrik K. Gustafsson, Martin Danelljan, and Thomas B. Schön. How reliable is your regression model's uncertainty under real-world distribution shifts?, 2023. URL https://arxiv.org/abs/2302.03679.

Xizewen Han, Huangjie Zheng, and Mingyuan Zhou. Card: Classification and regression diffusion models. *Advances in Neural Information Processing Systems*, 35:18100–18115, 2022.

James Hensman, Alex Matthews, and Zoubin Ghahramani. Scalable variational gaussian process classification. *arXiv preprint arXiv:1411.2005*, 2014.

Jose Hernandez-Lobato, Yingzhen Li, Mark Rowland, Thang Bui, Daniel Hernández-Lobato, and Richard Turner. Black-box alpha divergence minimization. In *International conference on machine learning*, pp. 1511–1520. PMLR, 2016.

Jiri Hron, Alexander G de G Matthews, and Zoubin Ghahramani. Variational gaussian dropout is not bayesian. *arXiv preprint arXiv:1711.02989*, 2017.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

Pavel Izmailov, Sharad Vikram, Matthew D Hoffman, and Andrew Gordon Gordon Wilson. What are bayesian neural network posteriors really like? In *International conference on machine learning*, pp. 4629–4640. PMLR, 2021.

Juan Emmanuel Johnson, Valero Laparra, and Gustau Camps-Valls. Accounting for input noise in gaussian process parameter retrieval. *IEEE Geoscience and Remote Sensing Letters*, 17(3): 391–395, 2019.

Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems*, 30, 2017.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Roger Koenker and Gilbert Bassett Jr. Regression quantiles. *Econometrica: journal of the Econometric Society*, pp. 33–50, 1978.

Ranganath Krishnan, Pi Esposito, and Mahesh Subedar. Bayesian-torch: Bayesian neural network layers for uncertainty estimation, January 2022. URL https://doi.org/10.5281/zenodo.5908307.

Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. Accurate uncertainties for deep learning using calibrated regression. In *International conference on machine learning*, pp. 2796–2804. PMLR, 2018.

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

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

David JC MacKay. A practical bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472, 1992.

Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. *Advances in neural information processing systems*, 32, 2019.

James Martens. New insights and perspectives on the natural gradient method. *The Journal of Machine Learning Research*, 21(1):5776–5851, 2020.

Alexander Graeme de Garis Matthews. *Scalable Gaussian process inference using variational methods*. PhD thesis, University of Cambridge, 2017.

Nis Meinert, Jakob Gawlikowski, and Alexander Lavin. The unreasonable effectiveness of deep evidential regression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 9134–9142, 2023.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

Kevin P Murphy. *Probabilistic machine learning: an introduction*. MIT press, 2022.

David A Nix and Andreas S Weigend. Estimating the mean and variance of the target probability distribution. In *Proceedings of 1994 ieee international conference on neural networks (ICNN'94)*, volume 1, pp. 55–60. IEEE, 1994.

Ian Osband. Risk versus uncertainty in deep learning: Bayes, bootstrap and the dangers of dropout. In *NIPS workshop on bayesian deep learning*, volume 192, 2016.

Kushagra Pandey, Avideep Mukherjee, Piyush Rai, and Abhishek Kumar. Diffusevae: Efficient, controllable and high-fidelity generation from low-dimensional latents. *arXiv preprint arXiv:2201.00308*, 2022.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Rajesh Ranganath, Sean Gerrish, and David Blei. Black box variational inference. In *Artificial intelligence and statistics*, pp. 814–822. PMLR, 2014.

Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable laplace approximation for neural networks. In *6th International Conference on Learning Representations, ICLR 2018-Conference Track Proceedings*, volume 6. International Conference on Representation Learning, 2018.

Yaniv Romano, Evan Patterson, and Emmanuel Candes. Conformalized quantile regression. *Advances in neural information processing systems*, 32, 2019.

Franko Schmähling, Jörg Martin, and Clemens Elster. A framework for benchmarking uncertainty in deep regression. *Applied Intelligence*, pp. 1–14, 2022.

Matthias Seeger. Gaussian processes for machine learning. *International journal of neural systems*, 14(02):69–106, 2004.

Mrinank Sharma, Sebastian Farquhar, Eric Nalisnick, and Tom Rainforth. Do bayesian neural networks need to be fully stochastic? In *International Conference on Artificial Intelligence and Statistics*, pp. 7694–7722. PMLR, 2023.

Laurens Sluijterman, Eric Cator, and Tom Heskes. Optimal training of mean variance estimation neural networks. *arXiv preprint arXiv:2302.08875*, 2023.

Michalis Titsias. Variational learning of inducing variables in sparse gaussian processes. In *Artificial intelligence and statistics*, pp. 567–574. PMLR, 2009.

Joost Van Amersfoort, Lewis Smith, Yee Whye Teh, and Yarin Gal. Uncertainty estimation using a single deep deterministic neural network. In *International conference on machine learning*, pp. 9690–9700. PMLR, 2020.

Joost van Amersfoort, Lewis Smith, Andrew Jesson, Oscar Key, and Yarin Gal. On feature collapse and deep kernel learning for single forward pass uncertainty. *arXiv preprint arXiv:2102.11409*, 2021.

Andrew G Wilson and Pavel Izmailov. Bayesian deep learning and a probabilistic perspective of generalization. *Advances in neural information processing systems*, 33:4697–4708, 2020.

Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P Xing. Deep kernel learning. In *Artificial intelligence and statistics*, pp. 370–378. PMLR, 2016.