CHEPAN: CONSTRAINED BLACK-BOX UNCERTAINTY MODELLING WITH QUANTILE REGRESSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Most of the predictive systems currently in use do not report any useful information for auditing their associated uncertainty and evaluating the corresponding risk. Taking for granted that their replacement may not be advisable in the short term, in this paper we propose a novel approach to modelling confidence in such systems while preserving their predictions. The method is based on the Chebyshev Polynomial Approximation Network (ChePAN), a new way of modelling aleatoric uncertainty in a regression scenario. In the case addressed here, uncertainty is modelled by building conditional quantiles on top of the original pointwise forecasting system considered as a black box, i.e. without making assumptions about its internal structure. Furthermore, ChePAN allows to consistently choose how to constrain any predicted quantile with respect to the original forecaster. Experiments show that the proposed method scales to large size data sets and transfers the advantages of quantile regression to estimating the black-box uncertainty.

1 INTRODUCTION

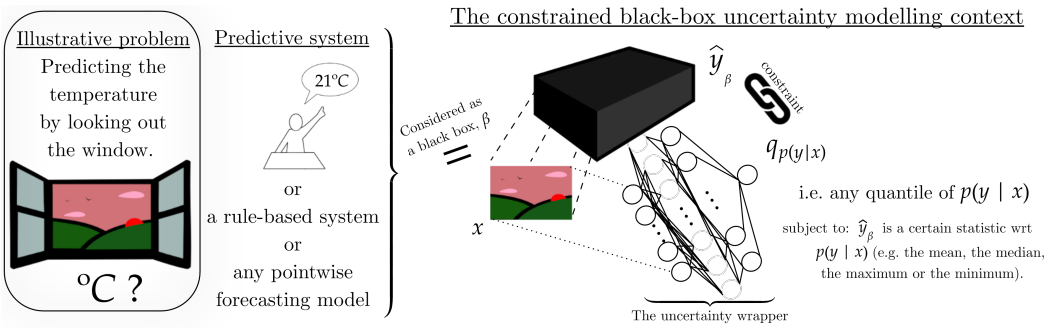


Figure 1: Description of the uncertainty modelling of a black box predictive system, β. This modelling is done by an uncertainty wrapper (the only neural network required part of the ChePAN model), which produces all the distribution $p(y \mid x)$ as quantiles, $q_{p(y|x)}$. ChePAN ensures that the original prediction of β corresponds to a desired statistic of $p(y \mid x)$, i.e. the constraint.

The present paper proposes a novel method for adding aleatoric uncertainty estimation to any pointwise predictive system currently in use. Considering the system as a *black box*, i.e. avoiding any hypothesis about the internal structure of the system, the methods offers a solution to the *technical debt* debate. The concept of *technical debt* was introduced in 1992 to initiate a debate on the long-term costs incurred when moving quickly in software engineering Sculley et al. (2015); Cunningham (1992). Specifically, most of the predictive systems currently in use have previously required much effort in terms of code development, documentation writing, unit test implementation, preparation of their dependencies or even their compliance with the appropriate regulations (e.g., medical Ustun & Rudin (2016) or financial models Rudin (2019) may have to satisfy interpretability constraints). However, once the system is being used with real-world problems, a new requirement can arise regarding the confidence of its predictions when the cost of an erroneous prediction is high. That being said, replacing the currently-in-use system may not be advisable in the short term. To address

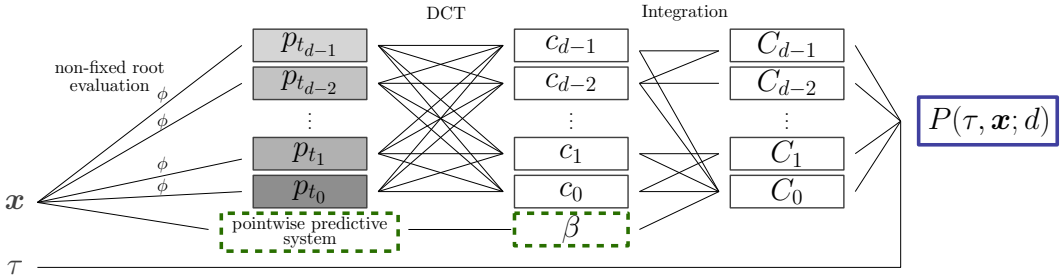


Figure 2: Graphical representation of ChePAN. For any degree d, $\{p_{t_i}\}_{i=0}^{d-1}$ are the evaluation of the initial Chebyshev polynomial expansion, $\{c_k\}_{k=0}^{d-1}$ their coefficients, $\{C_k\}_{k=0}^{d-1}$ the coefficients of the integrated polynomial, β the black box function and P the conditional prediction of the quantile τ.

this issue, the aim of this work is to report any information that is useful for auditing the system's associated uncertainty without modifying its predictions.

In general terms, sources of uncertainty can be understood by analysing the conditional members of this joint distribution: $p(y, \boldsymbol{x}) = \int_{\mathbb{M}} p(y \mid \boldsymbol{x}, M) p(M \mid \boldsymbol{x}) p(\boldsymbol{x})\, dM$ where $M \in \mathbb{M}$ is the family (assumed non-finite) of models that are considered.

Not all methods developed to model uncertainty can be applied in the black-box scenario, since the main hypothesis is that the black box is a fixed single model and internally unknown.

Specifically, those solutions that model *epistemic* uncertainty require to model $p(M \mid \boldsymbol{x})$. In other words, by epistemic we mean that uncertainty which can derive from ignorance about the model, including, for example, ensemble models Lakshminarayanan et al. (2017) or Bayesian neural networks Rasmussen (1996); Blundell et al. (2015); Hernández-Lobato & Adams (2015b); Teye et al. (2018) or MC-Dropout Gal & Ghahramani (2016). However, the black box could be a non-parametric predictive system or even a handcrafted rule-based system as it is shown in Figure 1. Hence the reason for studying aleatoric uncertainty, which originates from the variability of possible correct answers given the same input data, $p(y \mid \boldsymbol{x})$. This type of uncertainty can be tackled by modelling the response variable distribution. For instance, imposing a conditional normal distribution where the location parameter is the black-box function and the corresponding scale parameter is learnt. However, the more restricted the assumptions made about this distribution, the more difficult it will be to model heterogeneous distributions. One solution to this limitation is the type of regression analysis used in statistics and econometrics known as *Quantile Regression* (QR), which will provide a more comprehensive estimation.

Unlike classic regression methods, which only estimate a selected statistic such as the mean or the median, QR allows us to approximate any desired quantile. The main advantage of this method is that it allows confidence intervals to be captured without having to make strong assumptions about the distribution function to be approximated.

Recently, several works Dabney et al. (2018a); Tagasovska & Lopez-Paz (2018); Brando et al. (2019) have proposed a single deep learning model that implicitly learns all the quantiles at the same time, i.e. the model can be evaluated for any real value $\tau \in [0, 1]$ to give a pointwise estimation of any quantile value of the response variable. Nevertheless, these QR solutions are not directly applicable to the uncertainty modelling of a black box because the predicted quantiles need to be linked to the black-box prediction in some way.

In the present paper, we propose a novel method for QR based on estimating the derivative of the final function using a Chebyshev polynomial approximation to model the uncertainty of a black-box system. Specifically, this method disentangles the estimation of a selected statistic β of the distribution $p(y \mid \boldsymbol{x})$ from the estimation of the quantiles of $p(y \mid \boldsymbol{x})$ (shown in Figure 2). Hence, our method is not restricted to scenarios where we can jointly train both estimators, but can also be applied to pre-existing regression systems as a wrapper that produces the necessary information to evaluate aleatoric uncertainty. Additionally, the proposed method scales to several real world data sets.

This paper is organised as follows. Section 2 introduces the problem of QR and reviews the classic approach to use with neural networks and shows that it cannot be applied directly to the constrained black-box uncertainty modelling. Section 3 explores an approach to model the derivative of a function using neural networks. Both previous sections are the baseline to develop our proposed model and its properties, in Section 4. And finally, in Section 5, we show how our model can be applied in large size data sets and defines a new way to model the aleatoric uncertainty of a black box. The results are then summarised in the conclusion.

2 Conditional Quantile Regression

From the probabilistic perspective, solving a regression problem involves determining a conditional density model, $p(y \mid \boldsymbol{x})$. This model fits an observed set of samples $\mathcal{D} = (X, Y) = \left\{(\boldsymbol{x}_i, y_i) \mid \boldsymbol{x}_i \in \mathbb{R}^D, y_i \in \mathbb{R}\right\}_{i=1}^n$, which we assume to be sampled from an unknown distribution $q(y \mid \boldsymbol{x})$. In *Quantile Regression* (QR), we estimate q in a discrete manner by means of quantiles, which does not assume any typical parametric family distribution to the predicted p, i.e. it goes beyond central tendency or unimodality assumptions.

For each quantile value $\tau \in [0, 1]$ and each input value $\boldsymbol{x} \in \mathbb{R}^D$, the conditional quantile function will be $f\colon [0, 1] \times \mathbb{R}^D \to \mathbb{R}$. In our case, we use deep learning as a generic function approximator Hornik et al. (1989) to build the model f, as we shall see later. Consequently, f is a parametric function that will be optimised by minimising the following loss function with respect to their weights $\boldsymbol{w}$,

$$\mathcal{L}(\boldsymbol{x}, y, \tau) = \big(y - f_{\boldsymbol{w}}(\tau, \boldsymbol{x})\big) \cdot \big(\tau - \mathbb{1}[y < f_{\boldsymbol{w}}(\tau, \boldsymbol{x})]\big) \tag{1}$$

where $\mathbb{1}[c]$ denotes the indicator function that verifies the condition c. Equation 1 is an asymmetric convex loss function that penalises overestimation errors with weight τ and underestimation errors with weight $1 - \tau$.

Recently, different works Dabney et al. (2018b;a); Wen et al. (2017) have proposed deep learning models that minimise a QR loss function similar to equation 1. For instance, in the field of reinforcement learning, the Implicit Quantile Network (IQN) model was proposed Dabney et al. (2018a) and subsequently applied to solve regression problems as the Simultaneous Quantile Regression (SQR) model Tagasovska & Lopez-Paz (2019) or the IQN in Brando et al. (2019). These models consists of a neural network $\psi\colon [0, 1] \times \mathbb{R}^D \to \mathbb{R}$ such that it directly learns the function f that minimises equation 1, i.e. $f = \psi$. In order to optimise ψ for all possible τ values, these models pairs up each input $\boldsymbol{x}$ with a sampled $\tau \sim \mathcal{U}(0, 1)$ from a uniform distribution in each iteration of the stochastic gradient descent method. Thus, the final loss function is an expectation over τ of equation 1.

Despite the introduction of these QR models, **they cannot be applied to the constrained black-box scenario** given that they does not link their predicted quantiles with a pointwise forecasting system in a constrained way (Section 4.1). Other models, such as quantile forests, have a similar limitation. In the next section, we introduce the other main part needed to define our proposed method.

3 Modelling the Derivative with a Neural Network

Recently, a non-QR approach was proposed to build a monotonic function based on deep learning: the Unconstrained Monotonic Neural Network (UMNN) Wehenkel & Louppe (2019). The UMNN estimates the derivative of a function by means of a neural network, ϕ, which has its output restricted to strictly positive values, i.e. approaching $H(z)$ such that

$$H(z) = \int_0^z \phi(t)\, dt + H(0). \tag{2}$$

Therefore, if the neural network $\phi(z) \approx \frac{\partial H}{\partial z}(z) > 0$, this is in fact a sufficient condition to force $H(z)$ to be monotone.

To compute the integral of $\frac{\partial H}{\partial z}$, the UMNN approximates the integral of equation 2 using the Clenshaw-Curtis quadrature, which has a closed expression. The UMNN is designed to obtain a general monotonic function with respect to all the model inputs, z, but our interest is to build partial monotonic function with respect to the quantile value as we will explain hereafter.

4 CHEPAN: THE CHEBYSHEV POLYNOMIAL APPROXIMATION NETWORK

In this section, we will extend the UMNN to a model that is only monotonic with respect to the quantile input τ but not necessarily to $\boldsymbol{x}$. We will call this the Clenshaw-Curtis Network (CCN). Given that the quantile value works in a closed interval, $\tau \in [0, 1]$, we can use this fact to define our main proposal, the Chebyshev Polynomial Approximation Network (ChePAN), that additionally can transfer the advantages of quantile regression to the uncertainty modelling of a black box. We will develop the ChePAN model while comparing it with the CCN and describing the advantages of the former over the latter.

ChePAN and the CCN both contain a neural network $\phi\colon [0, 1] \times \mathbb{R}^D \to \mathbb{R}^+$ that only produces positive outputs and models the derivative of the final function with respect to τ. Both models optimise the neural networks $\phi(\tau, \boldsymbol{x})$ by calculating the coefficients of a truncated Chebyshev polynomial expansion $p(\tau, \boldsymbol{x}; d)$ of degree d with respect to τ but in a different manner[1].

Due to the aforementioned truncation in ChePAN, we consider a finite mesh of quantile values, called *Chebyshev roots*, $\{t_k\}_{k=0}^{d-1} \subset [0, 1]$ which are defined as

$$t_k = \frac{1}{2} \cos \frac{\pi(k + \frac{1}{2})}{d} + \frac{1}{2}, \quad 0 \leqslant k < d. \tag{3}$$

On the contrary, with the CCN, the Chebyshev roots are made to fit below the predicted quantile, τ, i.e. $\{t'_k\}_{k=0}^{d-1} = \tau \cdot \{t_k\}_{k=0}^{d-1} \subset [0, \tau]$. Thus, CCN's roots depend on τ and this implies avoiding the property of modelling the uncertainty of a black box as ChePAN has (described in Section 4.1).

The truncated Chebyshev expansion of a function can be interpreted as a linear transformation from the evaluations of ϕ on the roots, i.e. $\{\phi(t_k, \boldsymbol{x})\}_{k=0}^{d-1}$ into the Chebyshev coefficients $\{c_k(\boldsymbol{x})\}_{k=0}^{d-1}$, as Figure 2 illustrates. The linear transformation corresponds to the Discrete Cosine Transform of type-II (DCT-II), whose evaluation has complexity $\Theta(d \log d)$ rather than the classic matrix-vector product $\Theta(d^2)$.

The aforementioned coefficients $c_k(\boldsymbol{x})$ are those used in Chebyshev polynomial expansion, $p(\tau, \boldsymbol{x}; d)$, of $\phi(\tau, \boldsymbol{x})$ in the new basis of Chebyshev polynomials $T_k(t)$, i.e.

$$p(\tau, \boldsymbol{x}; d) = \frac{1}{2} c_0(\boldsymbol{x}) + \sum_{k=1}^{d-1} c_k(\boldsymbol{x}) T_k(2\tau - 1) \tag{4}$$

where $T_k(t)$ are defined recurrently as $T_0(t) = 1$, $T_1(t) = t$, and $T_{k+1}(t) = 2tT_k(t) - T_{k-1}(t)$ for $k \geqslant 1$. These polynomials T_k do not need to be explicitly computed. Note that, given the construction of the coefficients $c_k(\boldsymbol{x})$, the $p(t_k, \boldsymbol{x}; d)$ is equal to $\phi(t_k, \boldsymbol{x})$ in the root points t_k. That equality must be understood in terms of machine precision in the numerical representation system, classically $\sim 10^{-16}$ in double-precision or $\sim 10^{-8}$ in single-precision arithmetic. Figure 2 denotes this root evaluation step as p_{t_k}.

The goal is to provide $P(\tau, \boldsymbol{x}; d)$, which approximates $\int_0^\tau p(t, \boldsymbol{x}; d)\, dt$ and, in particular, also the integral of the neural network ϕ,

$$P(\tau, \boldsymbol{x}; d) \approx \Phi(\tau, \boldsymbol{x}) = \int_0^\tau \phi(t, \boldsymbol{x})\, dt + K(\boldsymbol{x}). \tag{5}$$

Due to the fact that we initially imposed that $\phi(\tau, \boldsymbol{x})$ must be positive for all $\tau \in [0, 1]$, then $P(\tau, \boldsymbol{x}; d)$ will be an increasing function with respect to τ. Since $p(\tau, \boldsymbol{x}; d)$ is a Chebyshev polynomial expansion, then its integral can be calculated using its coefficients $c_k(\boldsymbol{x})$ in equation 4.

According to Clenshaw (1955), the integral will give another Chebyshev expansion, $P(\tau, \boldsymbol{x}; d)$, with coefficients $C_k(\boldsymbol{x})$, i.e.

$$P(\tau, \boldsymbol{x}; d) = \frac{1}{2} C_0(\boldsymbol{x}) + \sum_{k=1}^{d-1} C_k(\boldsymbol{x}) T_k(2\tau - 1). \tag{6}$$

[1] Since the CCN is considered an extension of the UMNN, the supplementary material provides details of how to build this model.

Further, to deduce the expressions for $C_k(\boldsymbol{x})$ in equation 6, we need to recurrently integrate the polynomials $T_k(t)$, i.e.

$$\int T_0(t)\,dt = T_1(t) + \text{constant}, \qquad \int T_k(t)\,dt = \frac{T_{k-1}(t)}{2(k-1)} - \frac{T_{k+1}(t)}{2(k+1)} + \text{constant}. \tag{7}$$

By ordering the coefficients of the integral in Eq. equation 4, we deduce that

$$C_k(\boldsymbol{x}) \coloneqq \frac{c_{k-1}(\boldsymbol{x}) - c_{k+1}(\boldsymbol{x})}{4k}, \quad 0 < k < d-1, \qquad C_{d-1}(\boldsymbol{x}) \coloneqq \frac{c_{d-2}(\boldsymbol{x})}{4(d-1)}, \tag{8}$$

and $C_0(\boldsymbol{x})$ depends on the constant of integration $K(\boldsymbol{x})$ in equation 5 and the other coefficient values in equation 6.

On the one hand, the CCN considers the constant of integration as a separated bias. Therefore, this constant of integration will approximate the value of the first quantile $P(\tau = 0, \boldsymbol{x}; d)$.

On the other hand, ChePAN allows us to impose different conditions to obtain $C_0(\boldsymbol{x})$ for the full interval $[0, 1]$. This apparent downside will in fact become an advantage by giving rise to the next property of ChePAN as a solution for the black-box uncertainty modelling based on quantiles and discussed below.

4.1 Adding Uncertainty Estimation to a Black-Box Prediction System

In this subsection, we address situations in which there is a pointwise predictive system, which we will refer to as $\beta(\boldsymbol{x})$ and approximates a certain statistic such as the mean, median or a certain quantile of $p(y \mid \boldsymbol{x})$, as it is shown in Figure 1. This system is not required to be a deep learning model or even parametric. All that is needed to train the neural network ϕ in the ChePAN model is the corresponding β-evaluation values of the training set, i.e. $\{\boldsymbol{x}, \beta(\boldsymbol{x})\}$. Thus, our goal is to calculate the conditioned response distribution to the input without assuming asymmetry or unimodality with respect to this distribution as well as associating an statistic of this distribution to $\beta(\boldsymbol{x})$.

The formula used to calculate the constant of integration $C_0(\boldsymbol{x})$ will depend on which statistic we choose[2]. If we impose β to be the quantile $\tau = 0$ (which we shall call *ChePAN-β=q_0*), then

$$C_0(\boldsymbol{x}) = 2\beta(\boldsymbol{x}) - 2\sum_{k=1}^{d-1} C_k(\boldsymbol{x})(-1)^k. \tag{9}$$

However, if we force β to be the quantile $\tau = 1$ (which we shall call *ChePAN-β=q_1*), then

$$C_0(\boldsymbol{x}) = 2\beta(\boldsymbol{x}) - 2\sum_{k=1}^{d-1} C_k(\boldsymbol{x}). \tag{10}$$

For instance, the prediction of extreme weather events involves the forecasting system to predict the maximum or the minimum values of $p(y \mid \boldsymbol{x})$. In these cases, this pre-trained system could be used as β in equation 9 or equation 10, respectively, to determine the overall quantile distribution of $p(y \mid \boldsymbol{x})$ taking β as a reference point.

If β is the median (equivalently quantile $\tau = 0.5$) (which we shall call *ChePAN-β=Med*), then

$$C_0(\boldsymbol{x}) = 2\beta(\boldsymbol{x}) - 2\sum_{\substack{k=1 \\ k \text{ even}}}^{d-1} (-1)^{k/2} C_k(\boldsymbol{x}). \tag{11}$$

Finally, if β is the mean (which we shall call *ChePAN-β=Mean*), then

$$C_0(\boldsymbol{x}) = 2\beta(\boldsymbol{x}) - 2\sum_{\substack{k=1 \\ k \text{ odd}}}^{d-1} \frac{C_k(\boldsymbol{x})}{k^2 - 4}. \tag{12}$$

Additionally, $\beta(\boldsymbol{x})$ could be approximated by another neural network when we are optimising $\phi(\tau, \boldsymbol{x})$ and optimise both at same time. We will use this approach to compare ChePAN and other baseline models in the results section regarding black-box modelling.

[2] All details of how such formulas are reached can be found in the supplementary material.

Table 1: Mean and standard deviation, mean $\pm$ std, of the QR loss value of 10 executions of $\boxed{\text{Black box}}$-*Uncertainty wrapper* for the different test distributions of Figure 3 and three data sets.

	Asymmetric	Symmetric	Uniform	Multimodal	Year-MSD	BCN-RPF	YVC-RPF
$\boxed{RF}$-N	42.37 ± 0.04	23.19 ± 1.00	66.44 ± 0.26	151.51 ± 0.24	$57.50 \pm .05$	$23.47 \pm .14$	$\mathbf{27.27 \pm .39}$
$\boxed{RF}$-LP	42.88 ± 0.04	$\mathbf{22.10 \pm 0.03}$	67.13 ± 0.09	153.06 ± 0.22	$57.58 \pm .02$	$\mathbf{23.07 \pm .17}$	$28.06 \pm .12$
$\boxed{RF}$-ChePAN	$\mathbf{41.52 \pm 0.35}$	23.19 ± 0.70	$\mathbf{65.98 \pm 0.20}$	$\mathbf{148.39 \pm 0.16}$	$\mathbf{48.28 \pm .18}$	$\mathbf{23.17 \pm .07}$	$28.16 \pm .14$
$\boxed{XGBoost}$-N	$\mathbf{42.42 \pm 0.05}$	$\mathbf{23.35 \pm 0.99}$	66.38 ± 0.26	149.35 ± 0.40	$51.17 \pm .08$	$24.52 \pm .26$	$27.79 \pm .08$
$\boxed{XGBoost}$-LP	42.90 ± 0.02	$\mathbf{23.02 \pm 0.43}$	67.13 ± 0.17	150.94 ± 0.12	$51.24 \pm .02$	$22.63 \pm .11$	$27.86 \pm .07$
$\boxed{XGB.}$-ChePAN	$\mathbf{41.95 \pm 0.40}$	$\mathbf{23.69 \pm 0.68}$	$\mathbf{65.89 \pm 0.17}$	$\mathbf{146.20 \pm 0.30}$	$\mathbf{48.54 \pm .08}$	$\mathbf{22.00 \pm .04}$	$\mathbf{27.51 \pm .13}$
N	43.63 ± 2.89	$\mathbf{23.70 \pm 6.85}$	$\mathbf{67.45 \pm 1.68}$	148.78 ± 2.88	$49.00 \pm .24$	27.28 ± 1.25	28.62 ± 1.61
LP	43.46 ± 0.15	$\mathbf{20.72 \pm 0.47}$	$\mathbf{68.06 \pm 0.82}$	149.99 ± 0.64	$48.67 \pm .28$	$23.51 \pm .28$	$22.32 \pm .06$
ChePAN	$\mathbf{41.72 \pm 0.24}$	$\mathbf{22.94 \pm 1.81}$	$\mathbf{68.55 \pm 6.61}$	$\mathbf{145.93 \pm 3.14}$	$\mathbf{46.76 \pm .25}$	$\mathbf{20.67 \pm .40}$	$\mathbf{21.97 \pm .12}$

5 EXPERIMENTS

The source code used to reproduce ChePAN results in the following experiments can be found in the Github repository[3]. The DCT-II method referred to in Section 4 was used in the aforementioned source code.

In this section, we describe the performance of the proposed models compared to other baselines. The main goal is to show that using ChePAN, see Section 4.1, is an improvement compared to other black-box uncertainty modelling baselines, because it avoids centrality or unimodality assumptions just as QR does while it allows to choose how to constrain the predicted quantiles with respect to the black-box prediction.

5.1 MODELS UNDER EVALUATION

All exponential power distributions satisfy the condition that one of the parameters corresponds to the mode. Thus, these models that approximate such parametric distributions where the mode parameter is the black-box function and estimate the other parameter related to the uncertainty can be used as baselines.

- **The Heteroscedastic Normal distribution (N)** Similarly to Bishop (1994); Kendall & Gal (2017); Tagasovska & Lopez-Paz (2019); Brando et al. (2019) by using two neural networks, μ and σ, it can be approximated the conditional normal distribution, $\mathcal{N}(\mu(\boldsymbol{x}), \sigma(\boldsymbol{x}))$, such that they maximises the likelihood.

  In the black-box scenario proposed here, μ is the black-box function and we only need to optimise the σ neural network. Once optimised, the desired quantile τ can be obtained with $F(\tau, \boldsymbol{x}) = \mu(\boldsymbol{x}) + \sigma(\boldsymbol{x})\sqrt{2} \cdot \mathrm{erf}^{-1}(2\tau - 1)$, $\tau \in (0, 1)$, where erf^{-1} is the inverse error function.

- **The Heteroscedastic Laplace distribution (LP)** As a more robust alternative to outlier values, a conditional Laplace distribution, $LP\big(\mu(\boldsymbol{x}), b(\boldsymbol{x})\big)$, can be considered. Here, the quantile function is $F(\tau, \boldsymbol{x}) = \mu(\boldsymbol{x}) + \big(b \log(2\tau)\big) \cdot \mathbb{1}[\tau \leqslant \frac{1}{2}] - \big(b \log(2 - 2\tau)\big) \cdot \mathbb{1}[\tau > \frac{1}{2}]$, $\tau \in (0, 1)$.

- **The Chebyshev Polynomial Approximation Network (ChePAN)** In order to use the same black boxes than the other baselines, equation 12 is considered given that these black boxes are optimising the mean square error. In the pseudo code and in the Figure 6 of the supplementary material other alternative equations are considered.

5.2 DATA SETS AND EXPERIMENT SETTINGS

All experiments are implemented in TensorFlow Abadi et al. (2015) and Keras Chollet et al. (2019), running in a workstation with Titan X (Pascal) GPU and GeForce RTX 2080 GPU. All the details

[3]The camera-ready version of this paper will include all of the source codes to reproduce the experiments.

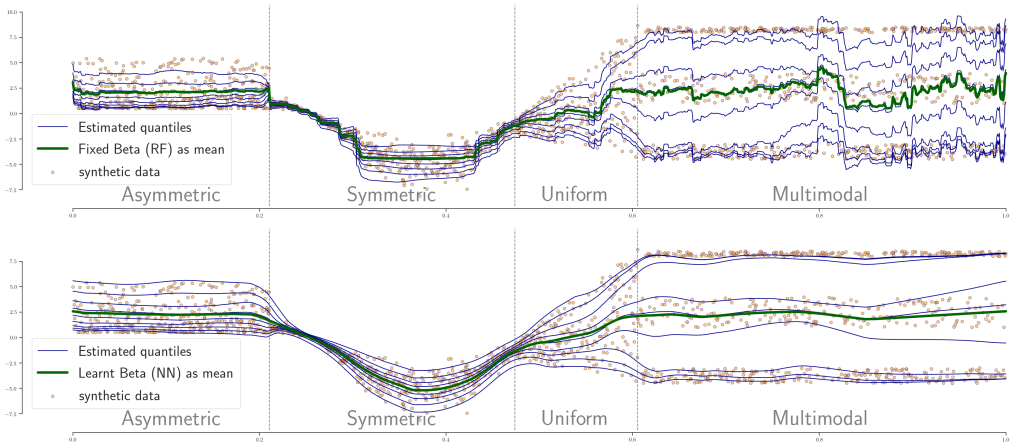


Figure 3: Heterogeneous synthetic distribution proposed by Brando et al. (2019). In the upper part of the figure, the learnt quantiles, ϕ, are noisy because their mean is the black box defined as an inaccurate MSE Random Forest (RF), β, following equation 12. In the lower part, ϕ and β are learnt and asymmetries and multimodalities can be seen more clearly, while still respecting the constraint in equation 12.

of the data sets used and model hyper-parameters for the results section are described in the supplementary material.

5.3 Results

Table 1 shows a comparison of uncertainty modelled for two given black-box systems (a Random Forest (RF) Liaw et al. (2002) and a XGBoost Chen & Guestrin (2016)) in four data sets. The first four columns correspond to each part of the synthetic distribution proposed by Brando et al. (2019) and shown in Figure 3, the fifth column is the full Year Prediction MSD UCI dataset Dua & Graff (2017a), to predict the release year of a song from 90 audio features and, finally, the last two columns correspond to predicting the room price forecasting of Airbnb flats (RPF) in Barcelona and Vancouver, extracted from Brando et al. (2019). The mean of the QR loss equation 1 value is evaluated for ten thousand randomly selected quantiles for ten executions of each model $\{m_k\}_{k=1}^{10}$,

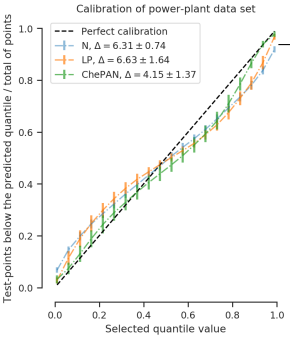
$$\mathcal{L}_{m_k}(X_{test}, Y_{test}) = \sum_{i=1}^{N_{test}} \sum_{j=1}^{N_\tau} \frac{\big(y_i - f_{m_k}(\tau_j, \boldsymbol{x}_i)\big) \cdot \big(\tau_j - \mathbb{1}[y_i < f_{m_k}(\tau_j, \boldsymbol{x}_i)]\big)}{N_{test} \cdot N_\tau}, \tag{13}$$

where N_{test} is the number of points in the test set, $N_\tau = 10,000$ the number of Monte Carlo samplings and f_{m_k} is each considered model of Table 1. Considering how the QR loss is defined in equation 1, its value not only informs us about the performance of each system but also how generically calibrated their predicted quantiles are.

Furthermore, in Table 1 we observe that ChePAN outperforms other methods in most cases due to it transferring the capacity to capture asymmetries and multimodalities of QR in $p(y \mid \boldsymbol{x})$ to the black-box problem, where we need to restrict our uncertainty modelling in order to maintain the corresponding statistic associated with the black box.

This restriction of conserving the black box can be seen qualitatively in the upper part of Figure 3, where the black box used is an inaccurate Random Forest predicting the mean. As it is shown, the quantiles predicted by ChePAN respect the noise of the black box while capturing the heterogeneous patterns of the distribution $p(y \mid \boldsymbol{x})$. On the other hand, the ability to model heterogeneous distributions using QR is better displayed in the lower part of the Figure 3. In that case, the black box is a neural network that is learnt concurrently with the quantiles. Since the black box is better fitted, the quantiles are better.

Finally, once we show that the performance order between the baselines is similar using the RF or the XGBoost, we want to show an ultimate experiments to directly measure the calibration of



	$XGBoost$-N	$XGBoost$-LP	$XGBoost$-ChePAN
Concrete	14.40 ± 1.7	14.35 ± 1.5	$\mathbf{5.99 \pm 4.4}$
Power	6.31 ± 0.7	$\mathbf{6.63 \pm 1.6}$	$\mathbf{4.15 \pm 1.4}$
Wine	$\mathbf{4.40 \pm 1.2}$	$\mathbf{4.34 \pm 1.2}$	$\mathbf{4.58 \pm 1.6}$
Yacht	$\mathbf{8.93 \pm 2.3}$	$\mathbf{6.98 \pm 2.4}$	14.20 ± 3.9
Naval	$\mathbf{6.61 \pm 2.8}$	$\mathbf{7.21 \pm 2.1}$	$\mathbf{6.90 \pm 2.7}$
Energy	$\mathbf{6.12 \pm 2.1}$	$\mathbf{5.05 \pm 2.0}$	$\mathbf{6.26 \pm 2.2}$
Boston	15.55 ± 3.1	15.30 ± 2.1	$\mathbf{4.99 \pm 1.7}$
Kin8nm	$\mathbf{3.55 \pm 0.6}$	4.79 ± 0.5	5.25 ± 0.8
Protein	7.88 ± 1.9	7.30 ± 0.3	$\mathbf{5.39 \pm 0.8}$

Figure 4: Plot with performance in terms of calibration. The table contains the mean and standard deviation of all the folds using the mean absolute error between the empirical predicted calibration and the perfect ideal calibration of 980 equidistant quantiles using equation 14.

the predicted quantiles and compare the predicted width of certain desired intervals. Following the UCI data sets used in Hernández-Lobato & Adams (2015b); Gal & Ghahramani (2016); Lakshminarayanan et al. (2017); Tagasovska & Lopez-Paz (2019), we have performed two empirical studies to assess this point in a black-box scenario where the black box is an MSE-XGBoost. Following the proposed hidden layers architecture in Tagasovska & Lopez-Paz (2019), the Prediction Interval Coverage Probability (PICP) and the Mean Prediction Interval Width (MPIW) are reported in the appendix Table 3 considering the 0.025 and the 0.975 quantiles. For the sake of completeness, in Figure 4 and its table we have also computed an additional metric not only to verify the calibration of the 0.025 and 0.975 quantiles, but also to obtain a measure of general calibration considering the entire quantile distribution. Given $\boldsymbol{\tau} = [10^{-2}, \ldots, 1 - 10^{-2}]$ N_τ-equidistant set of quantiles to evaluate, the % of actual test data that falls into each predicted quantile can be compared with each real quantile value as follows,

$$\mathcal{C}al(f; X_{test}, Y_{test}, \boldsymbol{\tau}) = \frac{1}{N_\tau} \sum_{j=1}^{N_\tau} |\tau_j - \frac{1}{N_{test}} \sum_{i=1}^{N_{test}} \mathbb{1}[y_i < f(\tau_j, \boldsymbol{x}_i)]| \tag{14}$$

Additionally, in appendix it can be found two extra figures of the disentangled visualisation of this calibration metric from each quantile. As all of the figures and tables show, in terms of calibration, ChePAN generally displays a better performance in the black-box scenario than the other models.

6 Conclusion

The uncertainty modelling of a black-box predictive system requires wrapper solutions to be designed that avoid assumptions about the internal structure of the system. Specifically, it could be a non-deep learning model (such as the one presented in Table 1 and Figure 3) or even a non-parametric predictive system, as it is proposed in Figure 1. Therefore, not all models or types of uncertainties can be considered using this framework.

The present paper introduces the Chebyshev Polynomial Approximation Network (ChePAN) model, which is based on Chebyshev polynomials and deep learning models and has a dual purpose: firstly, it predicts the aleatoric uncertainty of any pointwise predictive system; and secondly, it respects the statistic predicted by the pointwise system.

To conclude, ChePAN transfers the advantages of Quantile Regression (QR) to the problem of modelling aleatoric uncertainty estimation in another existing and fixed pointwise predictive system (denoted as β and called black box). Experiments with different large scale real data sets and a synthetic one that contains several heterogeneous distributions confirm these novel features.

REFERENCES

M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pp. 265–283, 2016.

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

Christopher M Bishop. Mixture density networks. 1994.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*, 2015.

Axel Brando, Jose A Rodriguez-Serrano, Jordi Vitria, and Alberto Rubio Muñoz. Modelling heterogeneous distributions with an uncountable mixture of asymmetric laplacians. In *Advances in Neural Information Processing Systems*, pp. 8836–8846, 2019.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016.

F. Chollet et al. Keras (2015), 2019.

C. W. Clenshaw. A note on the summation of Chebyshev series. *Math. Tables Aids Comput.*, 9: 118–120, 1955. ISSN 0891-6837.

Murray Cox. Inside airbnb: adding data to the debate. *Inside Airbnb [Internet].[cited 16 May 2019]. Available: http://insideairbnb.com*, 2019.

Ward Cunningham. The wycash portfolio management system. *ACM SIGPLAN OOPS Messenger*, 4(2):29–30, 1992.

W. Dabney, G. Ostrovski, D. Silver, and R. Munos. Implicit quantile networks for distributional reinforcement learning. In *International Conference on Machine Learning*, pp. 1104–1113, 2018a.

W. Dabney, M. Rowland, M. G. Bellemare, and R. Munos. Distributional reinforcement learning with quantile regression. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018b.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017a. URL `http://archive.ics.uci.edu/ml`.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017b. URL `http://archive.ics.uci.edu/ml`.

D. Elliott. Error analysis of an algorithm for summing certain finite series. *J. Austral. Math. Soc.*, 8: 213–221, 1968. ISSN 0263-6115.

Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. *arXiv:1506.02142*, 2015.

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, 2016.

J. M. Hernández-Lobato and R. Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *ICML*, pp. 1861–1869, 2015a.

José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International Conference on Machine Learning*, pp. 1861–1869, 2015b.

Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pp. 5574–5584, 2017.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pp. 6402–6413, 2017.

Andy Liaw, Matthew Wiener, et al. Classification and regression by randomforest. *R news*, 2(3): 18–22, 2002.

A. C. R. Newbery. Error analysis for polynomial evaluation. *Math. Comp.*, 28:789–793, 1974. ISSN 0025-5718. doi: 10.2307/2005700. URL `https://doi.org/10.2307/2005700`.

Carl Edward Rasmussen. A practical monte carlo implementation of bayesian learning. In *Advances in Neural Information Processing Systems*, pp. 598–604, 1996.

Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

David Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips, Dietmar Ebner, Vinay Chaudhary, Michael Young, Jean-Francois Crespo, and Dan Dennison. Hidden technical debt in machine learning systems. In *Advances in neural information processing systems*, pp. 2503–2511, 2015.

N. Tagasovska and D. Lopez-Paz. Frequentist uncertainty estimates for deep learning. *Bayesian Deep Learning workshop NeurIPS*, 2018.

Natasa Tagasovska and David Lopez-Paz. Single-model uncertainties for deep learning. In *Advances in Neural Information Processing Systems*, pp. 6417–6428, 2019.

Mattias Teye, Hossein Azizpour, and Kevin Smith. Bayesian uncertainty estimation for batch normalized deep networks. *arXiv preprint arXiv:1802.06455*, 2018.

Berk Ustun and Cynthia Rudin. Supersparse linear integer models for optimized medical scoring systems. *Machine Learning*, 102(3):349–391, 2016.

A. Wehenkel and G. Louppe. Unconstrained monotonic neural networks. In *Advances in Neural Information Processing Systems*, 2019.

R. Wen, K. Torkkola, B. Narayanaswamy, and D. Madeka. A multi-horizon quantile recurrent forecaster. *arXiv preprint arXiv:1711.11053*, 2017.

H. Zheng, Z. Yang, W. Liu, J Liang, and Y. Li. Improving deep neural networks using softplus units. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–4. IEEE, 2015.