university of groningen / faculty of science and engineering

Uncertainty estimation with neural network for time series data

Bachelor's Project Thesis

Levente Foldesi, s3980456, l.foldesi@student.rug.nl,
Supervisor: Dr M.A. Valdenegro Toro

Abstract: Nowadays, more and more high-stakes decisions are made using neural networks in order to make predictions. Specifically, meteorologists and hedge funds apply these techniques to time series data. When it comes to prediction, there are certain limitations for machine learning models (such as lack of expressiveness, vulnerability of domain shifts and overconfidence) which can be solved using uncertainty estimation. There is a set of expectations regarding how uncertainty should "behave". For instance, a wider prediction horizon should lead to more uncertainty or the model's confidence should be proportional to its accuracy. In this paper, different uncertainty estimation methods are used in order to forecast meteorological time series data and justify these expectations. The results show how each uncertainty estimation method performs on the forecasting task and confirm the expectations of uncertainty.

1 Introduction

In our modern civilization, neural networks are often used to tackle complex tasks. Many fields, such as robotics, computer vision, stock market/weather prediction, etc., use algorithms based on neural networks. Specifically, time series data with high volatility, rely heavily on neural networks when it comes to predicting future values. Nowadays, a great number of hedge funds and meteorologists use AI for different predictions. However, there are still plenty of limitations that need to be taken into account even when such a powerful tool is used. The most prominent issues described by Jakob et al. (2021) are the following:

- Neural networks are often called "black boxes" which suggests the lack of expressiveness and interpretability of the algorithm. In other words, it is hard to really understand every step that leads to the output, making it unreliable in some cases.
- Neural networks are vulnerable to domain shifts and struggle with identifying in-domain and out-of-domain data. This means that, for example, if an image classification model is trained to distinguish between cars and bikes, but during the testing a plane is given, the model will predict something instead of noticing the out-of-domain data.
- The output of a neural network does not provide uncertainty with it (how sure the given output is) and is often overconfident with its results.
- Neural networks are prone to adversarial attacks, which could cause them to malfunction.

The aforementioned limitations can be remedied (or at least improved upon) by estimating uncertainty for the outputs. Given these estimates, a human supervisor could decide whether to use the results of the network (at least parts of the results) or disregard them.

1.1 Uncertainty

There are two types of uncertainty which can be distinguished: aleatoric and epistemic. The former refers to uncertainty derived from the data itself (for instance, sensor noise) and the latter is uncertainty in the model, which can happen if the network is poorly designed or if there is a lack of data. Abdar et al. (2020) further discuss the introduced categories. Aleatoric uncertainty can have two further types: homoscedatic and heteroscedatic. The

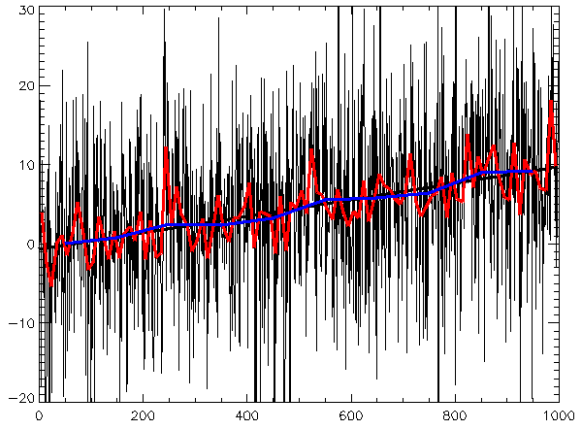

Figure 1.1: Example of a time series (Wikipedia, 2022).

aforementioned two terms indicate the σ^2 (variance) of the noise in a model. Homoscedatic means that σ^2 is constant and heteroscedatic means that σ^2 is a function of the input (thus varying for each different variable).

For the estimation of uncertainty, several techniques have been developed lately. In my research, I use Bayesian inference based and Ensemble based estimation techniques (Method section introduces them in detail).

1.2 Time-series data

Time-series data is a row of data points indexed by time. It is mainly used to track the change over time. Figure 1.1 presents and example.

Neural networks for time-series are usually used in order to predict future values (meteorological/stock prediction). Uncertainty can give validity to these predictions. For instance, a wider prediction horizon should have more uncertainty. Nevertheless, a standard point-wise-based network's prediction could be as confident for a narrow prediction horizon as for a wider one (Valdenegro-Toro, 2022).

1.3 State of art

The number of published papers on this topic has skyrocketed in the past few years. In this section I summarize the ones closely related to mine. Abdar et al. (2020) and Jakob et al. (2021) gave an overview of uncertainty in neural networks. Both papers described the mathematical/statistical backgrounds for the different approaches. Furthermore, they presented the state of the art on this topic. As stated earlier, both Bayesian and Ensemble methods are included in this paper. Gal (2016) issued a paper more specifically on Bayesian methods such as Monte Carlo Dropout, whereas Balaji et al. (2017) provided a paper on Ensemble methods, which is an easier alternative to Bayesian methods.

Siddique & Connor (2022) used Bayesian Neural Networks to classify auroral images into predefined labels and to predict the horizontal component of the perturbed magnetic field. These results were then compared to the results of a Gaussian Process Regression. The prediction experiment in this paper is fairly close to my research, which is described in the following section.

1.4 Proposed approach

This paper focuses on prediction for time series using different uncertainty methods. The papers mentioned above have given a theoretical overview of the different uncertainty methods and/or applied one or two of them to real-world data sets. This research focuses on applying several estimation techniques to two real world data sets. Comparing their results and analyzing uncertainty is the main aim of this paper.

There is a set of expectations for how uncertainty should behave when it comes to prediction for time series data.

1. A wider prediction horizon should result in higher uncertainty.
2. Larger errors (for instance, mean squared error) should yield higher uncertainty.
3. The estimation of calibration error (whether the model is over or under-confident). This suggests that the standard deviation (in other words, the uncertainty) should be proportional to the accuracy of the model.

I examine whether the aforementioned expectations are satisfied for time-series data using neural networks and the models described in section 2. Moreover, the results of different uncertainty quantification methods are compared.

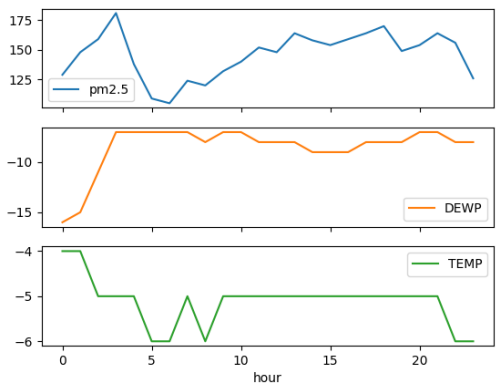

Figure 2.1: Example time series from the data set (PM2.5). It shows different values with a 24 hour range.

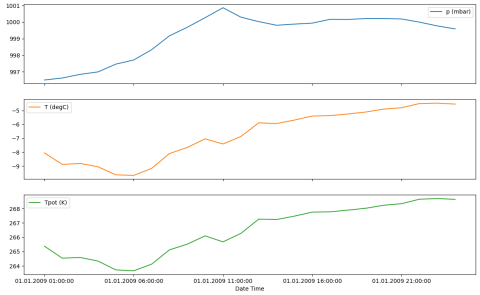

Figure 2.2: Example time series from the data set (Air pressure). It shows different values with a 24 hour range.

2 Methods

2.1 Data

Throughout the experiment, I used two meteorology data sets to test the different uncertainty models. The first is about PM2.5 concentration (it is referred to as **PM2.5 data set**) and it can be found here: `https://archive.ics.uci.edu/ml/datasets/Beijing+PM2.5+Data`. The data consists of 43825 time series with 8 features measured for each time frame (year, month, day and hour). The 8 features are the following: PM2.5 concentration, Dew Point, Temperature, Pressure, Combined wind direction, Cumulated wind speed, Cumulated hours of snow, and Cumulated hours of rain. The figure 2.1 above shows an example of 24 hours of data from PM2.5 concentration, Dew Point, and Temperature time series.

In this model, the prediction is made with regard to PM2.5 concentration. This is also called fine particulate matter, which has a harmful effect on human health (Pope & Dockery, 2006). For the predictions, the other values in the data set (features) are used as well (see the next section for more detail).

The second data set is a weather time series data (it is referred to as **Air pressure data set**). This data set can be downloaded here: `https://www.bgc-jena.mpg.de/wetter/`. In this data set, 14 different features were included: Air pressure, Air temperature, Potential temperature, Dew point temperature, Relative humidity, Saturation water vapor pressure, Actual water vapor pressure, Water vapor pressure, Deficit specific humidity, Water vapor concentration, Air density, Wind velocity, Maximum wind velocity and Wind direction. The data points are measured in every 10 minutes from 2009-2016. Since prediction regarding meteorological data is usually done by hours and the previous data set was also measured in each hour, the data set was converted from minute representation to hour representation by taking every 6^{th} data point only. Figure 2.2 shows some examples from the data set.

The prediction is done with respect to air pressure (in milibar). As mentioned before, regarding the features, a more comprehensive overview is given in the next section.

2.2 Preprocessing of the data

2.2.1 Cleaning and selecting the features which are used

After the data is imported, then the NaN (undefined or unrepresentable values) values are discarded since that could cause the neural network to break. Then a heat-map was used to decide which features are worth feeding to the model (Figure 2.3 shows the heat-map of the PM2.5 data).
Looking at figure 2.3, it is clear that there are no features which should not be included. Hence, the features which are used are Dew Point (the temperature needed to reach a relative humidity

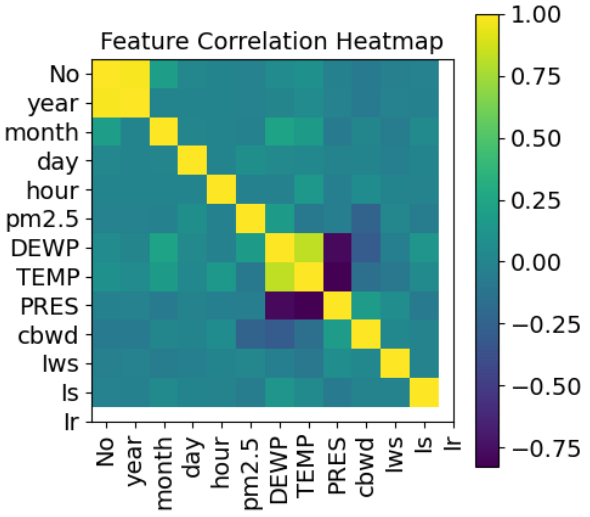

Figure 2.3: Heat-map of the PM2.5 data.

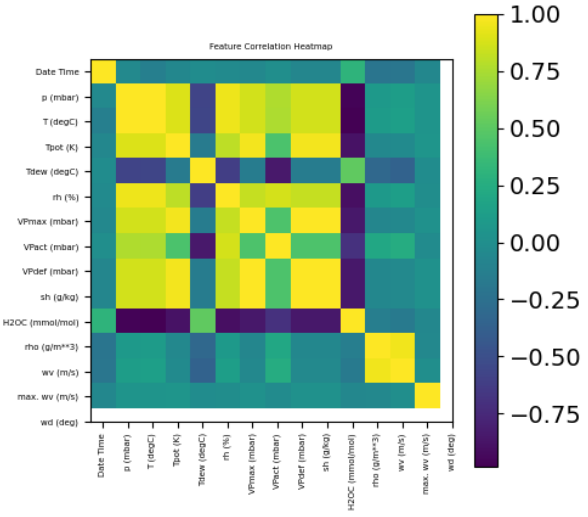

Figure 2.4: Heat-map of the Air pressure data.

of 100%), Temperature, Pressure, Cumulated wind speed, Cumulated hours of snow and Cumulated hours of rain. (Note that I do not include the Combined wind direction as it has a string data type). As a time-key, hours were used since weather related data can change rapidly. Thus, using the smallest unit available in the data set was the most applicable.

The heat-map for the second data set (Air pressure) is presented in figure 2.4. Here, it is clear that there are correlations amongst certain features consequently, the following were chosen for training: Air pressure, Air temperature, Saturation water vapor pressure, Specific humidity, Water vapor concentration, Wind velocity and Maximum wind velocity.

The further parts are identical for both data sets.

2.2.2 Normalization

Then one needs to normalize the training data (only the training data, the validation data remains untouched), since scaling is crucial for the network to work. This is done by subtracting the mean from each data point and then it is divided by its standard deviation. $\hat{D} = (d_i - \mu)/\sigma$ where $\hat{D}$ is the normalized data, d_i are the data points, μ is the mean and σ is the standard deviation.

Figure 2.5: Data windowing.

2.2.3 Making samples

Once the data is cleaned and the features are chosen, the data is split into training and validation sets. For that, a 0.715 splitting ratio was used. The next step is to create "x" and "y" values for both the validation and training data. These are the input (x), output (y) pairs.

Creating the samples was done by data windowing. The basic idea of this is that it divides both the training and validation data into input and label data (output). The input data are the "x" values and the label are the "y" values. It can either make the data for a single step prediction or a multiple-step prediction. The single step prediction is described in Figure 2.5.

.

The historical data is the input data and the tar-

get size shows how many hours ahead the algorithm predicts. The label data is used as a target (the y values). In this research, I use 120 data points are used as history, which is then sampled in a way that every 6^{th} data point is taken as an input. This suggests that the input contains 20 data points. The target size is 12 therefore, the 12^{th} value is used as an output (which the model will learn on). Hence, the model predicts a single value 12 hours ahead.
In order to check the uncertainty for a wider horizon, multi-step prediction is needed. This differs from the previous case in that not only the 12^{th} value is being predicted but every value up until the 12^{th} (so 12 predictions are made in total).

2.3 Models

The following sections describe all the models that were used for estimating uncertainty for predictions. For all methods, the representation of uncertainty is based on the estimated mean and variance. So, $U = [\mu - \sigma, \mu + \sigma]$ where μ is the predicted mean, σ is the predicted standard deviation, and U is the uncertainty presented with a confidence interval. The degree of uncertainty can be measured with the predicted standard deviation (σ).

2.3.1 Bayesian Neural Networks

As general neural network models cannot provide reliability for their predictions (as they only provide a point-wise prediction using, for instance, Maximum Likelihood Estimation) another method needs to be used. One method is called Bayesian Neural network where the main idea is to output a probability distribution instead of the point-wise weights. It deduces the probability distribution for the parameters in the network ($\lambda = (w_1, w_2, ..., w_K)$). This is done by applying the Bayes theorem on λ, given the training input/output pairs (x, y). The posterior distribution of the parameter space ($p(\lambda|x, y)$) is the actual learning algorithm, this makes the predictions. The assumed prior distribution on the weights is $p(\lambda)$ (Abdar et al., 2020). Then the bayesian equation is the following:

$$p(\lambda|x, y) = \frac{p(y|x, \lambda)p(\lambda)}{p(y|x)} \tag{2.1}$$

Since $p(y|x)$ is the probability which needs to be learned (the probability of some input/output pair), bayesian predictive posterior distribution can be calculated. Therefore, for a random test sample x^* the prediction can be modelled as follows:

$$p(y^*|x^*, x, y) = \int p(y^*|x^*\lambda)p(\lambda|x, y)d\lambda \tag{2.2}$$

This is called the Bayesian Model Averaging, which is the marginalisation of the likelihood with the posterior distribution (Jakob et al., 2021). This equation gives predictions for y^* with different parameters λ and weighs them by the probability of those parameters given the input x^*. The first part of the integral is called forward pass ($p(y^*|x^*\lambda)$) which calculates the probability distribution based on the weights and inputs. The second part ($p(\lambda|x, y)$) is called posterior of weights which is learned in the network using the Bayes rule presented above (Jakob et al., 2021). In general, this integral is hard to compute since the computation of $p(\lambda|x, y)$ (posterior distribution) cannot be done just by estimation. The reason for this is that neural networks rely on millions of weights/parameters which would mean the produced probability distributions have extremely high dimensions, hence it would be computationally expensive. The following models use different techniques to estimate the integral in 2.2.

2.3.2 Monte Carlo DropConnect/Dropout

As mentioned earlier, the integral in equation 2.2 cannot be calculated hence, it needs to be approximated, and for that, MC dropout is a powerful tool. The approximation is done by sampling, which, combined with Monte Carlo, produces a different sample from the posterior distribution after each forward pass. It overcomes the issue of slow computational time, which general Monte Carlo models suffer from. During training, it assigns random binary variables to certain units of the network, and if the value of a unit is 0, then the algorithm drops it. This prevents the layers from "co-tuning" too much (Abdar et al., 2020). Mathematically, the estimation is done as follows:

$$p(y^*|x^*, x, y) = M^{-1} \sum_{i}^{M} p(y^*|x^*\lambda)p(\lambda|x, y) \tag{2.3}$$

Then, the calculation of the mean and standard deviation is the following:

$$\mu(x) = \frac{1}{M}\sum_{i=i}^{M} f_i(x) \tag{2.4}$$

$$\sigma(x) = \sqrt{\frac{1}{1-M}\sum_{i=1}^{M}(f_i(x) - \mu(x))^2} \tag{2.5}$$

(Note that f_i represent the different samples and M is the number of forward passes.) From equation 2.4 and 2.5, the confidence interval is easily calculable. If the dropout is applied on weight instead of layers, then the method is called Monte Carlo DropConnect.

2.3.3 Bayes by Backprop (BBB)

Here, the computation of $p(\lambda|x, y)$ (posterior distribution) is done by estimation of a variational parameters, to get $q(\lambda)$ (a parametric distribution) closer to the posterior distribution, for that, one needs to use the Kullback-Leiber (KL) divergence (as a distance between $p(\lambda|x, y)$ and $q(\lambda)$) (Jakob et al., 2021):

$$KL(q(\lambda)||p(\lambda|x, y)) = \int q(\lambda) \log \frac{q(\lambda)}{p(\lambda|x, y)} d\lambda \tag{2.6}$$

One needs to minimise the divergence in order to get the predictive distribution however, since the posterior is unknown, the evidence lower bound (ELBO) needs to be maximised:

$$ELBO = \int q(\lambda) \log \frac{p(y|x, \lambda)}{q(\lambda)} \tag{2.7}$$

Then 2.7 can be rewritten as:

$$KL(q(\lambda)||p(\lambda|x, y)) = -ELBO + \log p(y|x) \tag{2.8}$$

This means the if ELBO is maximized then KL is minimised hence, the $q(\lambda)$ will be closer to the true posterior distribution (Abdar et al., 2020). The previous procedure is called variational approximation to the posterior distribution of the weights proposed by Hinton & Camp (1993). In practice, this means that each weight have a Gaussian distribution instead of a fixed number. Hence, the weights' parameters are $\mathcal{N}(\mu, \sigma)$ (which are updated by Gradient decent). (Note that since the weights are distributions, a stochastic gradient is needed; therefore, a Monte Carlo gradient is used). This can be achieved by minimizing the earlier mentioned KL divergence. This resulted in a variation free energy cost function (Charles et al., 2015):

$$\begin{aligned} F(\lambda|x, y) = KL[q(x, y, \lambda)||p(\lambda)] \\ - \mathbb{E}_{q(\lambda|x,y}[\log p(x, y, \lambda)] \end{aligned} \tag{2.9}$$

Here the KL term is for the divergence between the approximated posterior of the weights ($q(x, y, \lambda)$) and the prior ($p(\lambda)$). The term $-E_{q(\lambda|x,y}[\log p(x, y, \lambda)]$ is the loss of the model (Charles et al., 2015).

2.3.4 Flipout

The Flipout model is closely related to the aforementioned Bayes by Backrop (BBB) model. As previously described, BBB uses a Gaussian distribution for the weights hence the sampling process can be interpreted as a perturbation over the mean: $w = \mu + \sigma z$ where $z = \mathcal{N}(0, 1)$ and $w = \mathcal{N}(\mu, \sigma)$. Therefore, σz is an additive permutation to the mean, which causes randomness (Wen et al., 2018). This is improved upon by Flipout by reducing variance for the predicted distributions. Instead of a perturbation over the mean, a per-sample perturbation is used:

$$\Delta W_n = \Delta\hat{W} r_n s_n^T \tag{2.10}$$

Here ΔW_n is the standard deviation, $\Delta\hat{W} = \sigma z$ which is in this case the per-sample perturbation and r_n, s_n are independent samples from the Rademacher distribution (binary values, either -1 or 1). This method makes sure that each sample in a batch receives a different bias sample compared to BBB, where the samples vary only within each batch. This makes Flipout less noisy and faster (Wen et al., 2018).

2.3.5 Ensemble method

Ensemble methods are based on the "knowledge of the crowd" principle. This method is an easier alternative to Bayesian Neural Networks. The so-called ensemble members (different models) are making predictions, and the final prediction is the average of the ensemble members' outputs (Jakob et

al., 2021). This method deals with estimating the posterior distribution differently than the previous methods. Ensemble methods, as the name suggests, take the average of the results of the different models with different parameter settings. Similarly, uncertainty can be measured directly by taking the standard deviation of the ensembles' outputs. In mathematical terms:

$$\mu(x) = \frac{1}{M}\sum_{i=i}^{M} f_i(x) \tag{2.11}$$

$$\sigma(x) = \sqrt{\frac{1}{1-M}\sum_{i=1}^{M}(f_i(x) - \mu(x))^2} \tag{2.12}$$

Where $f(i)$ is the output of each ensemble member and M is the number of ensemble members. Then using the mean and the standard deviation, the confidence interval for the uncertainty is straightforwardly calculable.

2.3.6 Loss function

In all models (except the Baseline model, see section 2.4.1), the Mean Squared Error (MSE) was used as a loss function. It is a popular choice when it comes to regression. It punishes poor predictions sharply, which means that the outliers are vanished. This can be calculated as the following:

$$MSE(y, y^*) = \frac{1}{N}\sum_{i=1}^{N}(y_i - y_i^*) \tag{2.13}$$

Where y is the true value and y^* is the predicted value. In general, the lower MSE refers to a better result. A low MSE would indicate that the true mean is closer to the predicted mean therefore, the prediction is accurate.

2.4 Experiment design

This section describes the actual implementations and hyperparameters of each model. Moreover, the metrics are described for evaluation. Note that the whole experiment is done with both data set (PM2.5 and Air pressure).

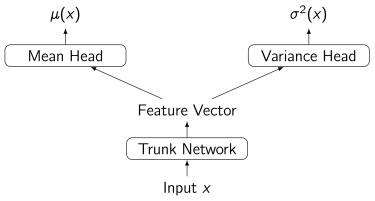

Figure 2.6: Model with two output heads (Valdenegro-Toro, 2022).

2.4.1 Baseline model

This model estimates uncertainty using a second output head of a neural network. This head produces the variance of the model. Since in this case, a direct, fairly simple, method was used to measure uncertainty, this model is used as a baseline for comparison.
Figure 2.6 shows the idea behind the two headed model. One head outputs the predicted mean and the other, the predicted variance. In order to estimate uncertainty, a special loss function is used, which is called Gaussian negative log likelihood (Note that all the other models use MSE as a loss since their uncertainty quantification is done within the neural network using special layers). This loss function increases the variance if the prediction is poor, which makes sure that poor performance produces large uncertainty.

$$-\log \mathbb{L}(y, \mu, \sigma^2) = \sum_i \log \sigma_i^2 + \frac{(y_i - \mu_i)^2}{\sigma_i^2} \tag{2.14}$$

If the prediction is poor $(y_i - \mu_i)^2$ is large hence σ_i^2 needs to be large as well in order to minimise the loss.

The baseline model consists of two hidden Dense layers with 32 neurons using the **relu** activation function (which is based on $max(x, 0)$). Since the data set is large, the "Adam" optimiser was used for the model. A learning rate of 0.001 was used. The number of epochs for training was 100. Figure 2.7 below shows the basic architecture of the network. The input layers have the shape of the training data. Then, as mentioned above, two Dense layers follow in the hidden layers and finally one Dense layer for the output.

This model was the basis for all the other different models mentioned before. In the following sections,

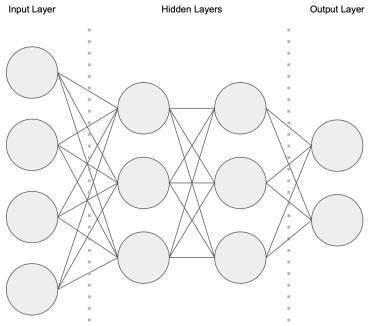

Figure 2.7: Architecture of the Baseline model.

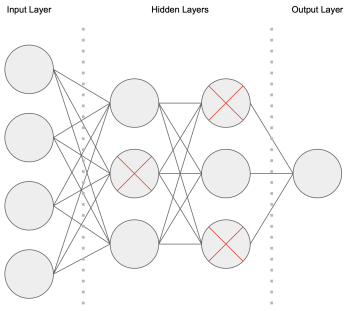

Figure 2.8: Architecture of the Dropout model.

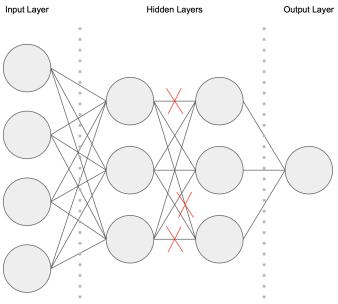

Figure 2.9: Architecture of the Dropconnect model.

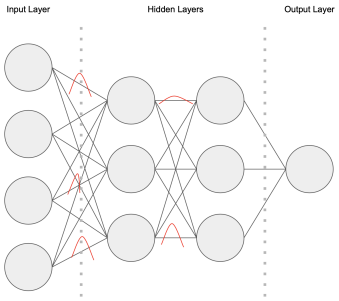

Figure 2.10: Architecture of the BBB model.

each model is described in terms of how they differ from the Baseline model (other than the loss function).

2.4.2 Dropout

The Dropout model consists of the same layers as the Baseline model however, a dropout with probability of 0.2 is applied on the hidden layers (using the **StochasticDropout** function from Valdenegro-Toro (2021)). This ensures that the units are dropped with the probability given above. Figure 2.8 below illustrates the network.

Then I used the **StochasticRegressor** function (Valdenegro-Toro, 2021) which give the predicted mean and standard deviation for the described model.

2.4.3 Dropconnect

Similarly as before, this model has the same architecture, however now the dense layers are switched to **DropConnectDense** (Valdenegro-Toro, 2021) which implements the dropping of the weights between the neurons. They take a parameter for the probability of dropping a weight. Figure 2.9 depicts the design of the model.
The dropping probability was 0.05. The other hyperparameters were the same as before. Then, again, the **StochasticRegressor** function (Valdenegro-Toro, 2021) was used to calculate the predicted mean and standard deviation.

2.4.4 Bayes by Backprop

Here, the dense layers are switched to **VariationalDense** (Valdenegro-Toro, 2021) which takes the parameters for the prior distributions. I used 5.0 and 2.0 for σ which is the variance of the Gaussian distribution. For π I used 0.5, which is for the Gaussian mixture distribution. The architecture can be seen in figure 2.10.

The number of epochs is 2500 since Bayes by Backprop requires several runs to converge.

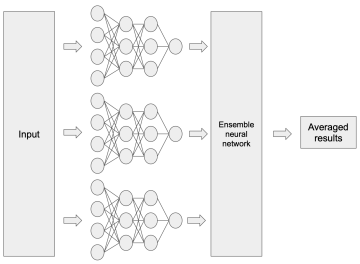

Figure 2.11: Architecture of the Ensemble model.

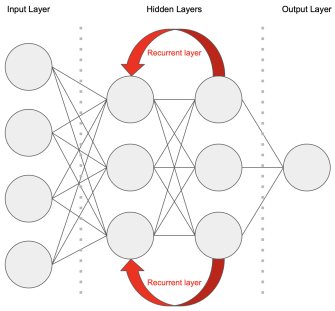

Figure 2.12: Architecture of the recurrent model.

2.4.5 Flipout

Flipout is a variation of the Bayes by Backprop model. Hence, the majority of the implementation is the same. However, **FlipoutDense** (Valdenegro-Toro, 2021) layer is used, which takes an additional argument (compared to BBB) which is bias distribution. It means that the biases are distributions and not scalars (Valdenegro-Toro & Saromo, 2022). Since it is a better version of BBB, fewer epochs are needed needed to converge; hence, 700 was used in this experiment.

2.4.6 Ensemble method

The architecture of the Ensemble model can be seen in figure 2.11. As explained above, this method averages the outputs of several models. The model consists of the same layers as the Baseline model. Then, to acquire the mean and the standard deviation, the **SimpleEnsemble** function was used (Valdenegro-Toro, 2021). It defines the number of models whose outputs then need to be averaged. In this experiment, it was 10.

2.5 Recurrent model

I have implemented each model, which was mentioned before, with an additional Long Short-Term Memory layer (LSTM) introduced by Hochreiter & Schmidhuber (1997). The architectures are the same as before just as a first layer, an LSTM is added. LSTM differs from regular layers in that it contains feedback connections, which makes it more powerful. Also, a gate is built in, through which the model "forgets" the irrelevant information, which makes it more memory efficient. Moreover, the relevant features can be stored for longer. Figure 2.12 shows the basic idea.

2.6 Metrics used for evaluation

2.6.1 Mean Absolute percentage error (MAPE)

MAPE is a metric that calculates the difference between the true value and the predicted value for each prediction, which is then divided by the true value. Then the differences are averaged. Note that the absolute value is crucial to prevent the positive and negative errors from canceling out.

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}|\frac{T_i - E_i}{T_i}| \tag{2.15}$$

Here n is the number of data points, T_i is the true value, and E_i is the prediction.

2.6.2 Mean squared error (MSE)

Equation 2.13 shows the calculation of the mean squared error. It can also be used as a metric to evaluate the performance of the model. In general, a lower MSE score is better.

2.6.3 R^2 score

R^2 score gives an insight into the difference between the true data and the predicted means. A large value suggests that the difference between them is small, hence the model fits well. Usually, R^2 is between 0 and 1, but a negative value is also possible. This suggests that the model is a poor fit for the

data. The calculation of R^2 is the following:

$$R^2(y, y^*) = 1 - \frac{\sum_{i=1}^{N}(y_i - y_i^*)^2}{\sum_{i=1}^{N}(y_i - \overline{y})^2} \tag{2.16}$$

Here, y^* is the predicted value, y is the true value, N is the number of data and $\overline{y}$ is the mean of the true value.

2.6.4 Calibration error

Calibration error is a measure between the confidence of the model and the accuracy of the model.

$$CE = \sum_i |acc(B_i) - conf(B_i)| \tag{2.17}$$

Here the confidence (probability between 0 and 1) is divided into equal parts, called bins B_i. For each bin, the accuracy is calculated (between 0 and 1 as well). Ideally, the model is as confident as accurate. So an error closer to zero is better (Valdenegro-Toro, 2022).

2.6.5 Negative log likelihood

The negative log likelihood evaluation method is popular for predictions with uncertainty. It is a proper scoring rule and can be applied for regression and classification as well (Lakshminarayanan et al., 2017). In this paper, a Gaussian assumption was used:

$$NLL = \frac{1}{n}\sum_{i=1}^{N}(\log(\sigma_i^2 + \epsilon) + \frac{(y_i - y_i^*)^2}{\sigma_i^2 + \epsilon}) \tag{2.18}$$

In equation 2.18, σ^2 is the predicted variance, y is the true mean, y^* is the predicted mean and ϵ is a constant. Furthermore, a lower score implies a better fit (Valdenegro-Toro, 2021).

3 Results

3.1 Qualitative analysis

The following section contains plots regarding how each model produced uncertainty and how well they forecasted the PM2.5 concentration or the Air pressure.

3.1.1 PM2.5

From figure 3.1 (left side) one can observe that poor predictions indeed produced large uncertainty, for instance, the fourth sample with the BBB or the third sample with Dropconnect. Conversely, when the prediction was closer to the real value, as in the second sample with the Flipout model, the model was more certain about its forecast. However, there are cases when the model is overconfident in its predictions. The first sample with the BBB model produced low uncertainty even though, the forecast was unreliable. In general, the Baseline model seems to perform the best in terms of the accuracy of the predictions. Consequently, the uncertainties are also rather low.

Figure 3.1 (right side) shows different samples which were chosen based on the mean squared error (see section 2.6.2). Here, the general expectation would be that the uncertainty is high since the difference between the actual mean and the predicted mean is large. However, one can observe that most models are overconfident in the samples provided in the graph. The Ensemble model seems to produce the most uncertainty regarding its predictions.

Figure 3.2 includes two different measures. The first row shows reliability plots, which indicate the confidence of the models. The x axis represents confidence whereas the y axis shows accuracy. In general, the model should be as accurate as confident so the perfect model would show a diagonal line. If the lines are below the diagonal, that means that the model is overconfident, if it is above, it is underconfident. As it can be seen, the vast majority of models seem to be overconfident (especially BBB and Flipout), except for the Baseline model. This is in line with the conclusions drawn from figure 3.1. Overconfidence can cause harm, as the model "believes" that its prediction is accurate even when it is not the case, leading to poor but certain decisions. The second row shows Error against Confidence plots. The idea is that a larger standard deviation (more uncertainty) should draw a larger error (in this paper the mean absolute error was used, a different metric would lead to different output). The plot displays both the testing and training data results. In most cases, the test data set seems to produce a desirable output, as a high standard deviation leads to a high error (except for the Baseline and Dropout models, where after an ini-

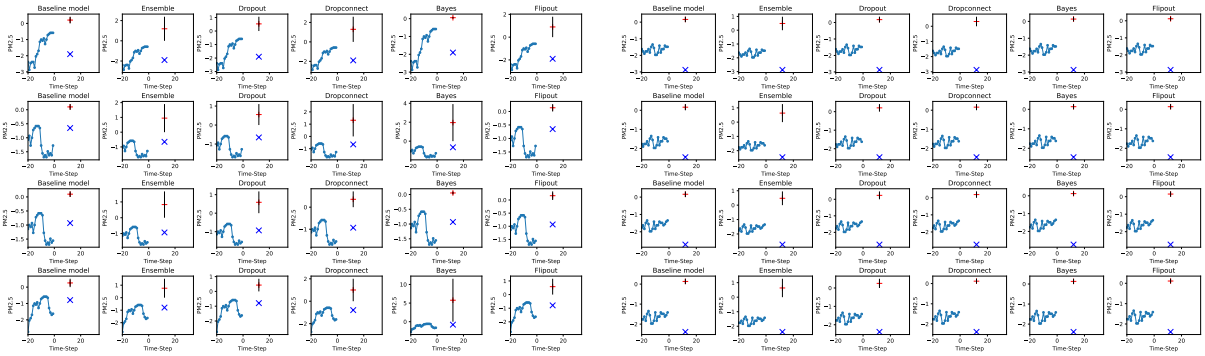

Figure 3.1: The graph shows predictions for each of the different non-recurrent models for the PM2.5 data set. The four "hardest" samples (time series) were chosen with respect to standard deviation, where large standard deviation means large uncertainty (on the left). Moreover, the four "hardest" samples with respect to mean squared error is presented as well (on the right). The blue cross shows the true value, the red small horizontal line represents the predicted mean, and the black vertical line is the confidence interval (uncertainty). Each row is a different sample, and each column is a different model.

tial rise, the error drops). The output of the training data seems to vary between methods. In general, it should produce a lower error as it has many more data points than the test data, and during the learning phase, the model aims to minimize the loss, so the error will be low. Note the two scores represented above the Confidence vs Error graphs. CE stands for calibration error, described in 2.17. It is analysed further in the next section (see tables 3.1 and 3.2 for the PM2.5 data and tables 3.3 and 3.4 for the Air pressure data). IS represents the Gaussian Interval score. It combines the wideness of the confidence interval (where the lower is better) and the coverage of the confidence interval which means how well the interval covers the true value. Hence, in general a lower IS score leads to a better model. From figure 3.2, it is clear that Ensembles has the lowest IS scores, closely followed by the Dropout and Baseline models. Flipout and BBB models have the largest IS scores, meaning that their uncertainty prediction is poor.

Figure 3.3 and 3.4 are based on the same idea as before but the outputs are produced with recurrent layers. In figure 3.3 (left side) it is clear how the Baseline model is underconfident. Most of its predictions are quite close to the true mean, and the uncertainty is still large. For instance, sample one was predicted fairly accurately by all models, and the most uncertainty was obtained with Baseline model. Sample four is a great example in the

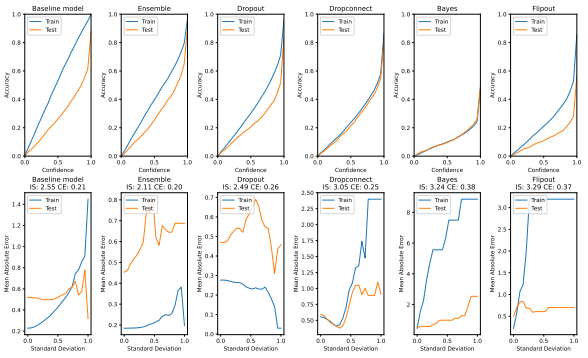

Figure 3.2: The graph shows the reliability plots (first row) and the Confidence vs Error plots (second row) for the PM2.5 data set, without recurrent layers.

sense that it shows how the further the true mean is from the predicted mean (further with Dropconnect and closest with Flipout) the larger the confidence interval gets.

Comparing the recurrent models to the standard ones, it seems that there are fewer cases when the uncertainty is extremely large using LSTM. Regarding the accuracy of the predictions, the recurrent models seems to predict closer to the true value.

Figure 3.3 (right side) is similar to the results acquired with the standard models. As seen before, the Ensemble model produced the largest uncertainty however, all models are still overconfident. In section 3.2, more details are given regarding the comparison of the two types of models (recurrent vs standard).

Comparing figure 3.4 to the results obtained for the standard model, notice that the Flipout and BBB models have a bit worse reliability. Furthermore, Dropconnect seems to be almost perfectly calibrated using LSTM layers. The other models are similar.

The Confidence vs Error plots for the recurrent models are much more coherent. Almost all models produce larger uncertainty with larger error. Only the Dropout model's error drops with increasing uncertainty after a short rise. Dropconnect model's error fluctuates a bit as well. BBB, Flipout, Ensemble and the Baseline model produced a bit better results than their standard versions. Regarding the IS scores, Ensembles still has the best performance however, Dropconnect model's score is significantly better than before. Moreover, notice how BBB model produced 7.10 which around twice as large as before.

In order to check whether wider prediction horizon yields higher uncertainty, multi-step prediction is needed (as described in the methods section). This way, the model predicts 12 hours ahead one by one. Figure 3.5 shows how uncertainty changes amongst the predictions on the horizon. It is clear that uncertainty does not increase when the model predicts further into the future. In many cases, the uncertainty does get higher for a few steps, but then it drops (last two samples with Dropout model). In most cases, it is hard to find a pattern between uncertainty and the prediction horizon. In the next section (3.2), different metrics are used as well to evaluate the this behaviour.

Figure 3.6 shows the same results as before, just for the recurrent models. The results seem to justify the prediction horizon expectation a bit better. For instance, the Baseline model clearly shows that the $10^{th} - 12^{th}$ predictions have larger uncertainty compared to the earlier ones. Furthermore, sample four with the Dropconnect model seems to produce higher uncertainty with a larger prediction horizon (even though the increment is not gradual). In many other cases, such as with the Dropout model, the first few predictions have higher uncertainty, but then from the 8^{th} prediction till the end, the uncertainty increases gradually.

3.1.2 Air pressure

In this section, the second data set is evaluated, similarly as the previous one. Firstly, figure 3.7 (left side) shows the samples with the highest uncertainty, and how each model predicted the true mean and the standard deviation. The first sample (first row) shows perfectly how the closer the prediction gets to the true value, the smaller the confidence interval gets. However, this is not always the case, for example, for the third sample (third row) the Bayes by Backprop model's prediction was almost identical to Flipout's prediction, but it yielded a lower uncertainty, which suggests that the Bayes model is overconfident (this was the case with the previous data set as well). Overall, it seems that the Bayes by Backprop and the Flipout model produces the narrowest confidence intervals hence, they seem to be the most overconfident (again in line with the previous findings). Moreover, later on with the reliability plots and the calibrated error, this is confirmed.

Figure 3.7 (right side) shows the samples with the largest error (MSE). In this example, one can observe some interesting behaviours. The first sample (first row) shows some cases when the models are underconfident. Baseline, Ensemble and Dropout models produced high uncertainties even though the prediction was close to the true value. Another interesting sample can be seen in row three. Here, despite the previously seen examples, only BBB predicted large uncertainty spite of the fact that all other model's predictions were poor. This somehow contradicts the previous findings but note that since this figure only shows a few examples, no conclusive conclusion can be drawn directly.

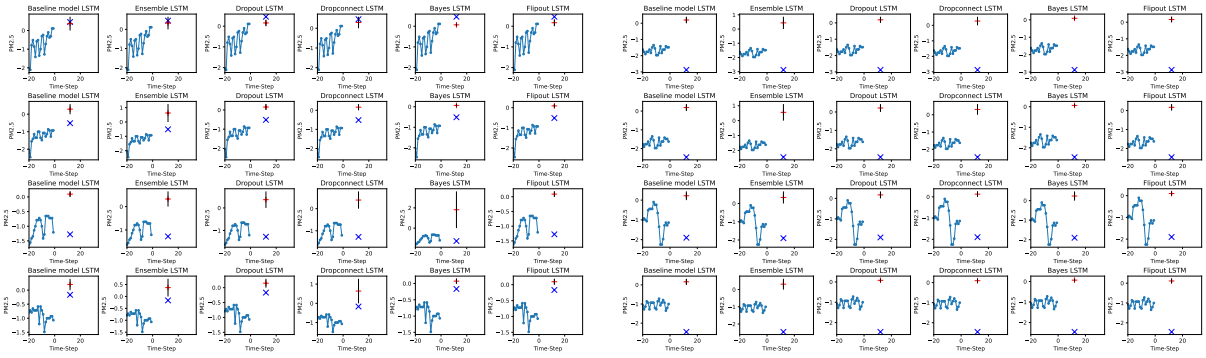

Figure 3.3: This figure, similarly to 3.1, shows four predictions for different models with large uncertainties but, using recurrent layers. So the left side takes the largest uncertainties and the right side takes the highest means squared error.

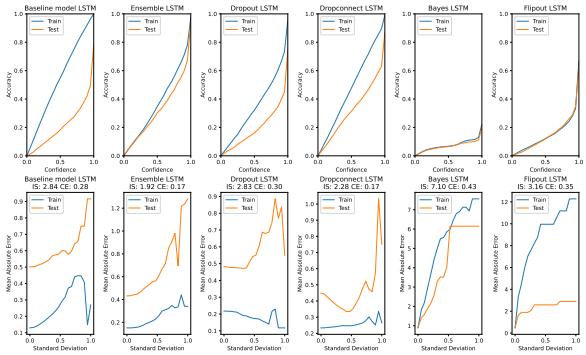

Figure 3.4: Reliability (first row) and Confidence vs Error (second row) plots, using recurrent layers for PM2.5 data.

The reliability plots (3.8) are similar to the one obtained before. The Ensemble and Dropout models seem to produce the best results. Dropconnect seems to be a bit more overconfident than with the PM2.5 data set. Regarding the confidence vs error plots, only Baseline Dropout and BBB models are behaving as expected. For the other models the uncertainty does not get higher for larger error. For instance, for the Flipout model the error drops gradually. The Ensebmle model produced large error up until 0.8 standard deviation, but then it suddenly plummets close to zero. Also note how for the BBB model, the training data also shows similar behaviour as the testing data. The reason behind it, that ideally, if the model learns the data it should have low error score for all the predictions; however, if the model does not learn efficiently the error stays large by the end of the training. Consequently, the BBB model learns poorly thought the process hence, the training data will have a large error as well as the uncertainty increases. For the IS score, very similar results were obtained to the PM2.5 data set's scores. The rank of the model is unchanged.

The following plots (3.9 and 3.10) depict the same result as before, only with LSTM layers. The results obtained are fairly similar. In 3.9 (left side) there is a case, when the model predicts perfectly (BBB model, second sample). Note how the uncertainty was almost zero in that case. In figure 3.9 (right side) it is clear that BBB and Flipout are way too overconfident (except for sample four with BBB where it predicts a large confidence interval). Moreover, the second sample causes trouble for all models in a sense that their predictions are off, and the predicted standard deviations are almost zero. In figure 3.10, just as with the PM2.5 data set, Dropconnect seems to be better calibrated. Furthermore, Dropout and the Baseline model are calibrated significantly worse. For the second row, note how the LSTM layer, with this data set, causes almost all models to confirm the expectation of higher error yield higher uncertainty. The only outlier is the Dropconnnect model where the error drops around 0.8 standard deviation. Another significant difference is that the largest MAE, in this case, for BBB is 18 and for Flipout is 60 whereas without LSTM it was 20 for BBB and one for Flipout. This suggest that in this scenario, BBB may outperform the Flipout model. For the IS scores, as before, the

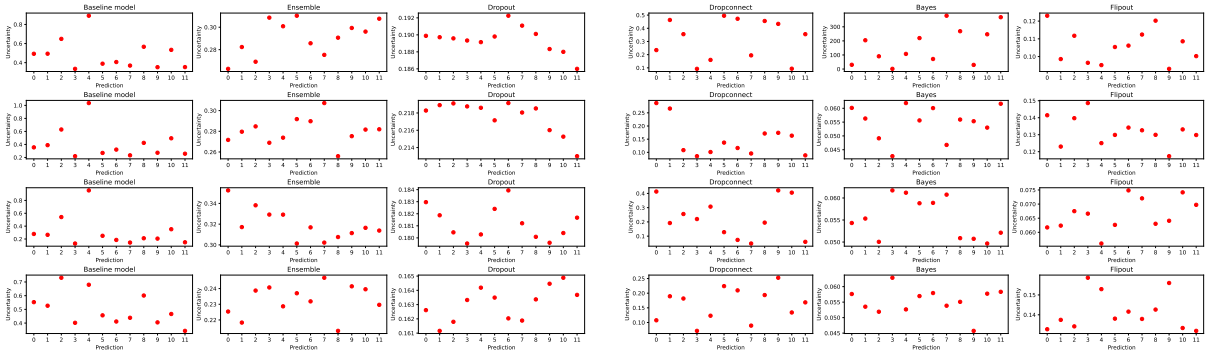

Figure 3.5: Uncertainty for wider horizon for models without recurrent layers. Each row shows a different sample and each column shows a different method. The results show how uncertainty does not always increases with wider horizon. (These results are for the PM2.5 data set.)

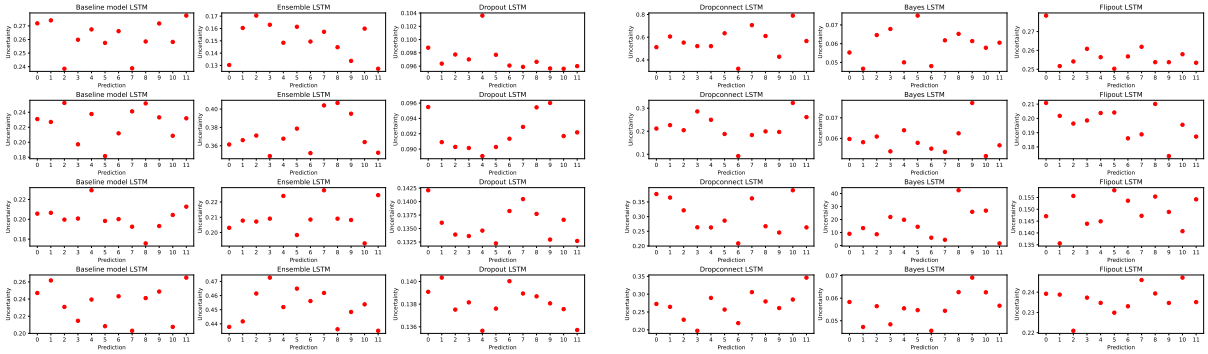

Figure 3.6: Uncertainty for wider horizon with recurrent layers. Each row shows a different sample and each column shows a different method. In this setting, the Baseline model seems to produce higher uncertainty for predictions more into the future. (These results are for the PM2.5 data set.)

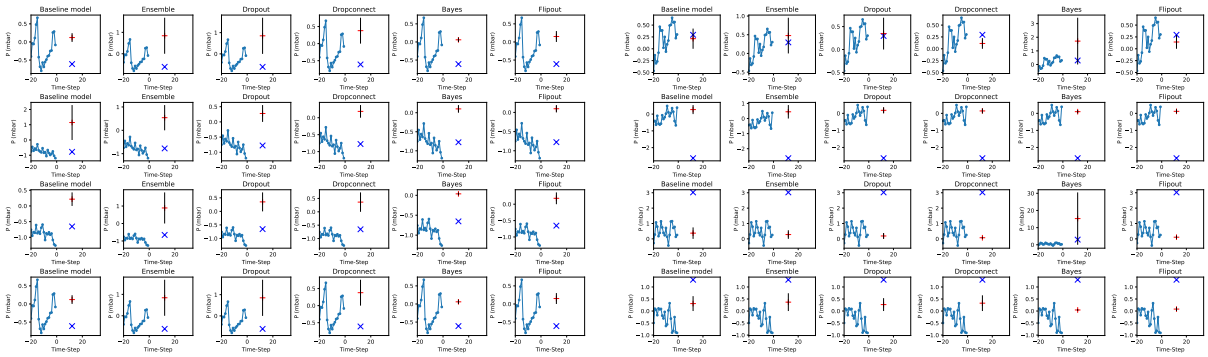

Figure 3.7: Samples with the highest uncertainty (left) and with the highest mean squared error (right) for the Air pressure data set without recurrent layers.

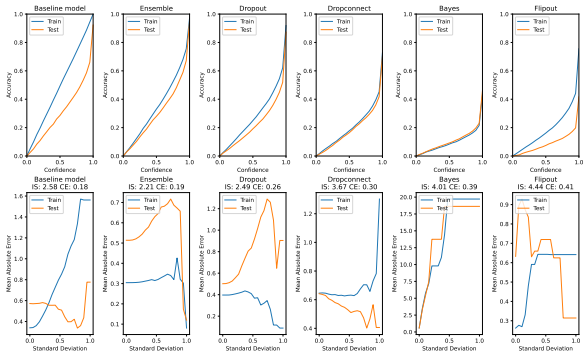

Figure 3.8: Reliability (first row) and Confidence vs error (second row) plots for the Air pressure data.

rank of the model stays unchanged compared to the PM2.5 data set's score. In this next section a quantitative analyses is done to acquire more robust results.

Figure 3.11 represents the multi-step predictions for the Air temperature data set. Ensemble and Dropout models seem to produce perfect results for the first two samples as, almost step by step, the uncertainty gets higher. Dropout model performs the opposite way for the third and fourth sample, as one would expect. The first few predictions have the largest uncertainty and the later ones have less. The Baseline model, although not gradually, but also seems to produce lower and lower uncertainty for future predictions. The other three models seem to have a random pattern.

Similarly as before, the prediction horizons are checked but with LSTM layer (3.12. In this case, the results change significantly. Baseline model seems to predict uncertainty increasingly for all the steps into the future. The Ensemble model works as expected for the first, whereas the Dropout model works starting from the $4^{th} - 5^{th}$ step. In some cases, Dropconnect does predict higher uncertainty for values more into the future, in most cases, it does not have a pattern. Flipout and BBB is almost identical to the previous results.

3.2 Quantitative analysis

In this section, the analysis of the aforementioned metrics is presented. The following two tables contain the summary of the metrics for the PM2.5 data set. Table 3.1 contains the result of the models without a recurrent network, and table 3.2 contains the one with LSTM layers.

Models/Metrics	MAPE ↓	MSE ↓	R^2 ↑	Calibration error ↓	NLL ↓
Baseline	51.85	0.45	0.56	0.21	5.57
Ensemble	**45.29**	**0.36**	**0.65**	**0.20**	**1.86**
Dropout	46.99	0.38	0.62	0.26	4.38
Dropconnect	59.14	0.52	0.49	0.25	5.52
BBB	47.00	0.37	0.64	0.37	29.71
Flipout	50.14	0.43	0.57	0.35	17.84

Table 3.1: Metrics scores of each model (PM2.5).

Table 3.1 depicts the results of the models without recurrent layers. The Ensemble model performed the best according to all metrics. Flipout and Baseline models were the worst fit for the data, but the scores were rather similar, there were no outliers. It is also worth mentioning how Flipout and BBB performed worse in uncertainty related metrics (CE and NLL) compared to other models. Nevertheless, especially BBB, fitted the data well in the first three metrics. This suggests that while BBB is a good model in terms of prediction accuracy, regarding uncertainty estimation, it is fairly poor.

Models/Metrics	MAPE ↓	MSE ↓	R^2 ↑	Calibration error ↓	NLL ↓
Baseline (lstm)	50.16	0.46	0.55	0.28	8.04
Ensemble (lstm)	**43.13**	**0.34**	**0.67**	**0.17**	**1.61**
Dropout (lstm)	48.21	0.41	0.60	0.29	8.59
Dropconnect (lstm)	44.78	0.36	0.65	0.17	4.33
BBB (lstm)	92.48	1.15	-0.11	0.42	102.71
Flipout (lstm)	50.14	0.41	0.59	0.34	13.85

Table 3.2: Metrics scores of each model with recurrent layer (PM2.5).

Table 3.2 shows the metrics for the models using recurrent networks. The results are somewhat similar as the best algorithm is still the Ensemble model but Dropconnect with recurrent layers seems to perform much better (this was supported by the reliability plots). Furthermore, BBB performed extremely poorly, with a negative R^2 score and an MSE larger than 1. Also, the NLL for BBB is extremely poor which suggests that it predicts uncertainty poorly. The other models produced similar and accurate results.

Amongst all the models, Ensemble performed the best, with a slight margin ahead of Dropout or Dropconnect depending on the recurrent layers. Both the standard and recurrent versions of these

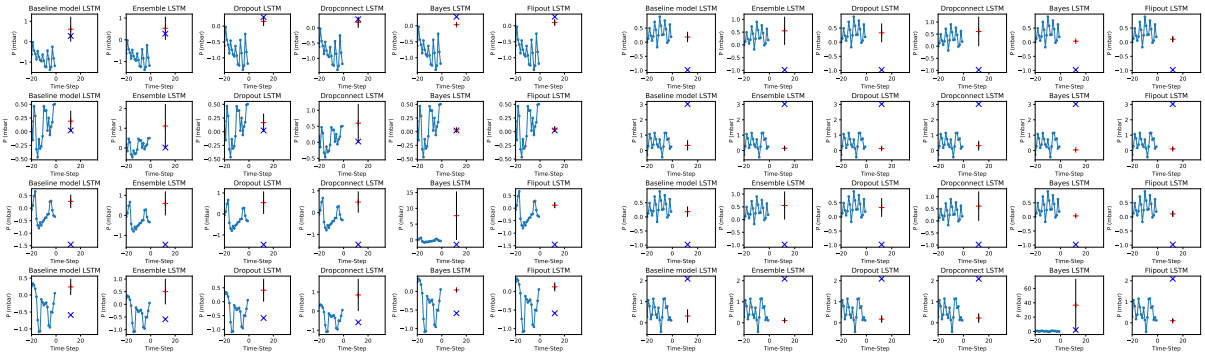

Figure 3.9: Samples with the highest uncertainty (left) and with the highest mean squared error (right) for the Air pressure data set with LSTM.

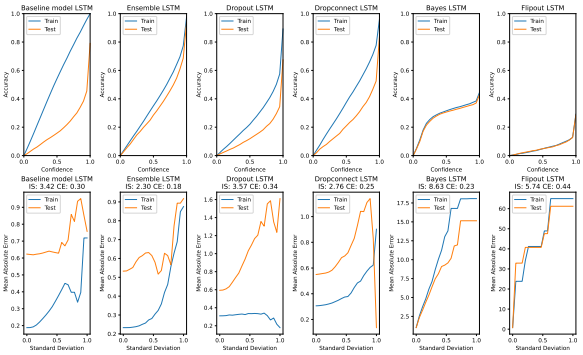

Figure 3.10: Reliability (first row) and Confidence vs error (second row) plots for the Air pressure data with LSTM.

models outperformed the other estimation methods.

For the Air pressure data set, the following two tables contains the produced results: 3.3 and 3.4. The results differ from the PM2.5 data set. In this setting, the best model was not straight forward as before. Without LSTM, the best model, in regards with predicting the true mean, was the Dropout model. As before, Ensemble was the best in predicting uncertainty, since it produced the lowest NLL score. Interestingly, the Baseline model was the best calibrated in a tie with Ensemble, which means that they were the least overconfident amongst the models. Note how in this case, both BBB and Flipout performed a bit worse than for the PM2.5 data set. With the recurrent layer, the best model was Ensembles again furthermore, as seen with the PM2.5 data set, Dropconnect preformed significantly better with LSTM. Both Flipout and BBB have really low performance both in predicting the true mean and the standard deviation. Note that BBB had really low calibration error compared to the previous cases.

In general, the models performed worse with the Air pressure data set than with the PM2.5 data set. It is also worth mentioning that the Baseline model performed relatively well in all settings, especially with CE and NLL scores. Overall the best model for uncertainty was Ensemble however, in some cases, Dropout or Dropconnect outperformed it.

Models/Metrics	MAPE ↓	MSE ↓	R^2 ↑	Calibration error ↓	NLL ↓
Baseline	57.19	0.53	0.36	0.18	2.62
Ensemble	51.37	0.42	0.49	**0.18**	**1.32**
Dropout	**50.18**	**0.40**	**0.51**	0.26	3.07
Dropconnect	64.17	0.62	0.25	0.30	18.84
BBB	57.56	0.65	0.21	0.38	38.70
Flipout	63.25	0.69	0.17	0.40	36.16

Table 3.3: Metrics scores of each model (Air pressure).

	MAPE ↓	MSE ↓	R^2 ↑	Calibration error ↓	NLL ↓
Baseline (lstm)	62.31	0.62	0.25	0.29	6.70
Ensemble (lstm)	**53.26**	**0.46**	**0.45**	**0.17**	**1.56**
Dropout (lstm)	59.34	0.57	0.32	0.34	10.51
Dropconnect (lstm)	54.96	0.48	0.42	0.25	4.04
BBB (lstm)	101.22	2.18	-1.60	0.23	196.70
Flipout (lstm)	75.99	2.54	-2.04	0.43	96.60

Table 3.4: Metrics scores of each model with recurrent layer (Air pressure).

As mentioned earlier, a quantitative analysis was done for checking the expectations with regard to the prediction horizons. Figure 3.13 represents different metrics for each time-step on the prediction

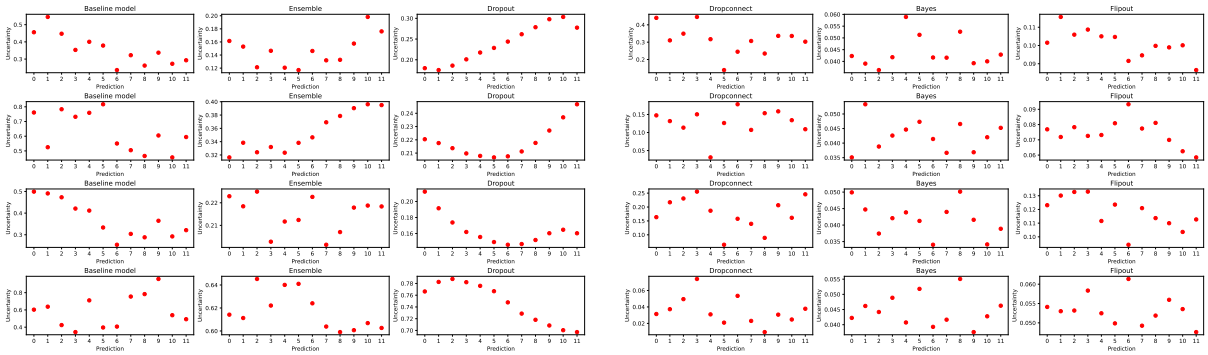

Figure 3.11: Uncertainty for wider horizon without recurrent layers. Each row shows a different sample and each column shows a different method. In this setting, the Ensemble and Dropout models seems to work the best. (These results are for the Air pressure data set.)

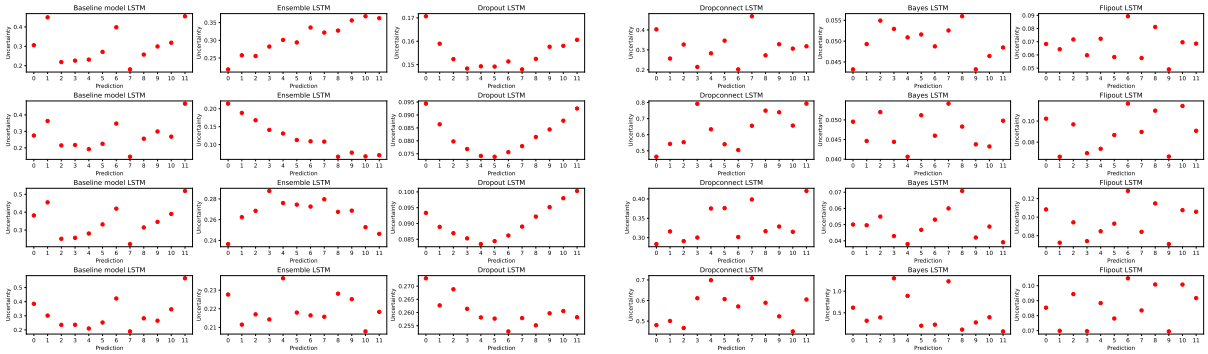

Figure 3.12: Uncertainty for wider horizon with recurrent layers. Each row shows a different sample and each column shows a different method. Here, Ensemble and Baseline models seems to "behave" the most desirable. (These results are for the Air pressure data set.)

horizon for the PM2.5 data. MAPE, MSE, CE, NLL should be larger when the model predicts more into the future whereas R^2 should be lower. As seen previously, this is not always the case. In some cases the MAPE, MSE scores are increasing but then there is a sudden drop around the $4^{th} - 7^{th}$ prediction, then the scores start to rise again. The R^2 only seems to drop for the Baseline model for the further predictions. With the other models, the score either fluctuates or rise. For the Baseline model, the NLL seems to be worse and worse but not gradually, whereas for Ensmeble and Dropout, it decreases in the beginning and then ascends at the furthest predictions. The Flipout model also seems to behave as expected regarding NLL as, more or less gradually, the score climbs. Regarding the CE score, only the BBB model seems to produce desired results as the other models' CE fluctuates. Figure 3.14 represents the same results as the aforementioned figure (3.13) but for the LSTM models. Similarly as before, in some cases the expectations are fulfilled however, here in the majority of the cases, it is hard to find a consistent pattern.

The same results for the Air pressure data set are presented in figure 3.15 and 3.16. 3.15 shows the results for the non-recurrent version. Here the Dropout model seems to produce the expected results perfectly as all of the scores are increasing as the model predicts more into the future and the R^2 is decreasing with a larger prediction horizon. The Ensemble model produced larger CE for future values, which is in line with the previous findings as with all the data sets, Ensemble performed the best regrading calibration. The rest of the models are either performing poorly (BBB and FLipout) or randomly (Dropconnect). This is again supported by the results as BBB and FLipout, in general, performed badly. 3.16 represents the same results but with LSTM layer. Notice that the Dropout model works a bit worse compared to the non-recurrent version. The predictions more in the future yield worse scores. The Baseline model seems to work the opposite way with the all the metrics, which makes sense as it has the simplest way predicting uncertainty, hence it is possible that it causes trouble to it. The Ensemble model performs somewhat correctly for uncertainty related metrics (NLL and CE) which is in line with the previous findings (the reliability plots showed that the Ensemble model was fairly well calibrated). Dropconnect is expected to perform better as with LSTM, it produced better results in other metrics. However, other than being less noisy, it does not produce the required results. The Flipout and BBB model still performs poorly and noisily.

4 Discussion

In this section, a summary of the paper is given together with ideas for future research. In order to summarize the findings of this experiment table 4.1 and 4.2 ranks the different models in all the investigated measures of uncertainty with time series. Table 4.1 shows all the models for the PM2.5 data set. Firstly, the metrics used for regression are discussed. In this setting, the Ensemble, Ensemble with LSTM and Dropconnect with LSTM models are the strongest. The worst models are Dropconnect and BBB with LSTM. The calibration of the models produced similar results. The same three models are found to be the best however, the BBB models (both with and without LSTM) are the worst in terms of calibration. For predicting uncertainty (NLL), as before, Ensemble, Ensemble with LSTM and Dropconnect with LSTM models are the most powerful. Moving forwards, in the research question, it was stated that a wider prediction horizon should yield larger uncertainty (and worse predictions in general). This was measured with three categorical parameters: Bad, Moderate and Good.

Bad where there were no conclusive results.

Moderate when there were at least two metrics from figure 3.13 and 3.14 (3.15 and 3.16 for the Air pressure data) which satisfied the expectations.

Good if more than two expectations are satisfied.

For this data set, only the Baseline and Ensemble models produced the desired output. The vast majority of the models did not have a clear pattern as the amplitude of uncertainty was randomly changing on the prediction horizon. Regarding the Confidence vs Error expectation, the LSTM models performed better in general. In this case:

Bad means that the error does not increase with larger uncertainty.

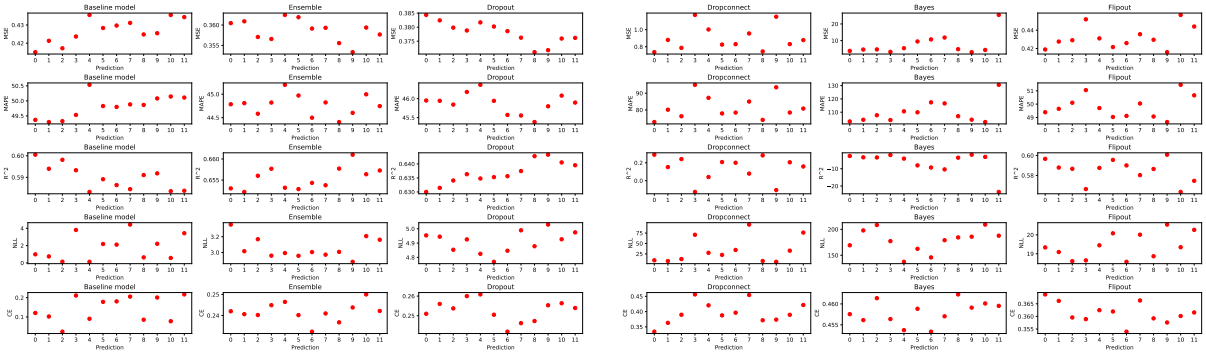

Figure 3.13: Five different metrics (MAPE, MSE, R^2, CE, NLL) to evaluate the prediction horizon (without recurrent layers for the PM2.5 data set). As seen in the previous section (3.1), the scores not always follow a gradual rise (or drop in case of R^2).

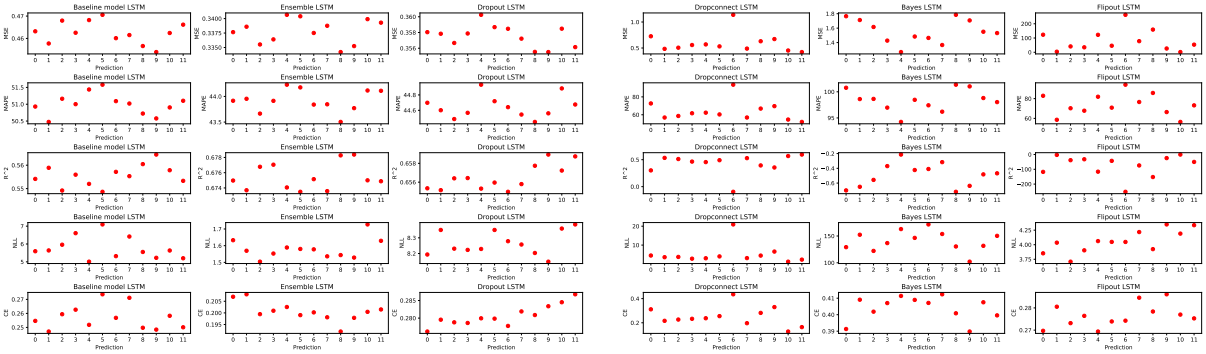

Figure 3.14: Five different metrics (MAPE, MSE, R^2, CE, NLL) to evaluate the prediction horizon (with recurrent layers for the PM2.5 data set). Theses results are less conclusive than for the non-recurrent models.

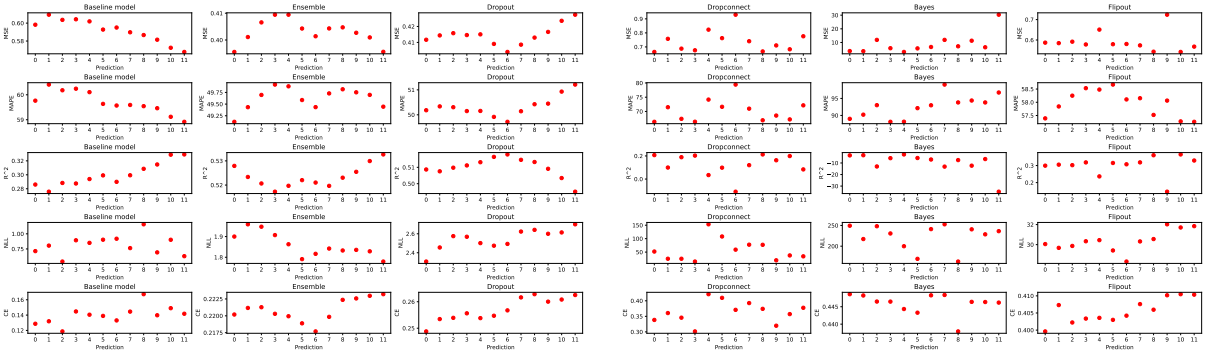

Figure 3.15: Five different metrics (MAPE, MSE, R^2, CE, NLL) to evaluate the prediction horizon (without recurrent layers for the Air pressure data set). In this setting, Dropout models seems to produce the desired results for all the metrics.

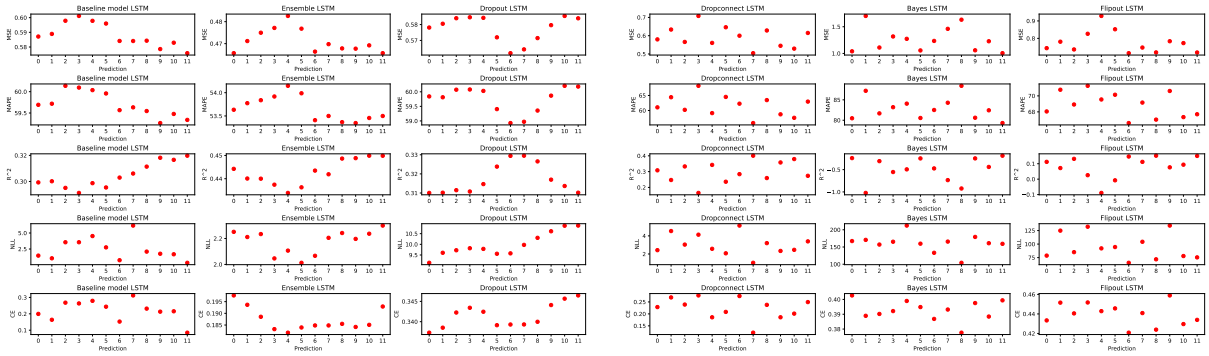

Figure 3.16: Five different metrics (MAPE, MSE, R^2, CE, NLL) to evaluate the prediction horizon (with recurrent layers for the Air pressure data set). Similarly as with the non-recurrent setting, the Dropout model "behaves" the closest to the expectations.

Moderate means that the error of the testing data increases but only with 15-20% or it fluctuates but with higher uncertainty it gets higher (or the error of training data gets very high as well).

Good means that the error increases with more than 20% and the increment is does not oscillate.

The best performance was obtained by the Baseline with LSTM and Dropout with LSTM models (Note that figure 3.2 and 3.4 were used for PM2.5 data and 3.8 and 3.10 for the Air pressure data). The non-recurrent models' errors either decreased gradually with more uncertainty or fluctuated heavily.

Table 4.2 represents all the models for the Air pressure data. For the standard metrics (MAPE, MSE and R^2) the best three models were Ensemble, Dropout and Ensemble with LSTM. The worst performances were achieved by BBB with LSTM and Flipout with LSTM. For the calibration error, the Baseline model outperformed the Dropout model hence it shares the podium with the two Ensemble models. In this case, the Flipout models produced the highest calibration error. For the NLL score, the best three models are the same as for the previous case. The BBB and Flipout models (both with and without LSTM) were poor fit for this task. For the horizon and the Confidence vs Error expectations, the same measures were used as for the PM2.5 data set. First, for the prediction horizon, the Dropout models seem to perform the best, while the Ensemble models are either inconclusive or partly correct. Then finally, for the Confidence vs Error expectation, as for the other data set, the LSTM models performed better especially, the Baseline, Ensemble and Dropout models.

To reflect on the research questions, the expectations of uncertainty when it comes to time series data are fulfilled, but the choice of model is crucial. In general, Ensemble and Dropout/Dropconnect models performed the best. However, the selection of the model is highly dependent on the data set and the task. The data set and the model also influence the need for the recurrent layers. For example, they outperformed the standard models in Confidence vs Error expectation but the horizon expectation worked better for the non-recurrent models. Hence, for future research, different versions of Ensemble and Dropout/Dropconnect models could be experimented with to perfect uncertainty estimation for time series.

References

Abdar, M., Pourpanah, F., Hussain, S., Rezazadegan, D., Liu, L., Ghavamzadeh, M., ... Nahavandi, S. (2020). A review of uncertainty quantification in deep learning: Techniques, applications and challenges.
doi: https://doi.org/10.1016/j.inffus.2021.05.008

Balaji, L., Alexander, P., & Charles, B. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles.

Data	PM2.5						
Models/Metrics	MAPE	MSE	R^2	Calibration error	NLL	Horizon	Confidence vs Error
Baseline	10	9	9	4	6	**Good**	Bad
Ensemble	**3**	**3**	**3**	**3**	**2**	**Good**	Moderate
Dropout	4	5	5	6	4	Moderate	Bad
Dropconnect	11	11	11	5	5	Bad	Moderate
BBB	5	4	4	11	11	Moderate	Bad
Flipout	7	8	8	10	10	Bad	Bad
Baseline (lstm)	9	10	10	7	7	Bad	**Good**
Ensemble (lstm)	**1**	**1**	**1**	**1**	**1**	Bad	**Good**
Dropout (lstm)	6	6	6	8	8	Bad	Moderate
Dropconnect (lstm)	**2**	**2**	**2**	**2**	**3**	Bad	Moderate
BBB (lstm)	12	12	12	12	12	Bad	Moderate
Flipout (lstm)	8	7	7	9	9	Bad	Moderate

Table 4.1: The table represents the ranking of models regarding the different metrics/tasks for the PM2.5 data. The Ensemble models and the LSTM version of the Dropconnect model performed the best. For the horizon check, only the Baseline model performed in the desired manner. Regarding the Confidence vs Error plots, the LSTM Baseline and Ensemble models performed as expected.

Data	Air pressure						
Models/Metrics	MAPE	MSE	R^2	Calibration error	NLL	Horizon	Confidence vs Error
Baseline	5	5	5	**3**	**3**	Bad	Moderate
Ensemble	**2**	**2**	**2**	**2**	**1**	Moderate	Bad
Dropout	**1**	**1**	**1**	6	4	**Good**	Moderate
Dropconnect	10	8	8	8	8	Bad	Bad
BBB	6	9	9	10	10	Moderate	Moderate
Flipout	9	10	10	11	9	Moderate	Bad
Baseline (lstm)	8	7	7	7	6	Bad	**Good**
Ensemble (lstm)	**3**	**3**	**3**	**1**	**2**	Bad	**Good**
Dropout (lstm)	7	6	6	9	7	**Good**	**Good**
Dropconnect (lstm)	4	4	4	5	5	Bad	Bad
BBB (lstm)	12	11	11	4	12	Moderate	Moderate
Flipout (lstm)	11	12	12	12	11	Bad	Moderate

Table 4.2: The table represents the ranking of models regarding the different metrics/tasks for the Air pressure data. The Ensemble models and the Dropout model performed the best in general. For the uncertainty based metrics, the Baseline model was successful as well. Regrading the horizon check, the Dropout models significantly outperformed all the other model. Regarding the Confidence vs Error plots, the LSTM Baseline, Ensemble and Dropout models performed as expected.

doi: https://arxiv.org/pdf/1612.01474.pdf

Charles, B., Julien, C., Koray, K., & Daan, W. (2015). Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*.

Gal, Y. (2016). Uncertainty in deep learning. *Ph.D. dissertation, University of Cambridge*.

Hinton, G., & Camp, D. V. (1993). Keeping the neural networks simple by minimizing the description length of the weights. *Proceedings of the 16th Annual Conference On Learning Theory (COLT)*, 5-13.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

Jakob, G., Cedrique, M., Rovile N.T. andAlih, & Jung, P. (2021). A survey of uncertainty in deep neural networks.
doi: https://doi.org/10.48550/arXiv.2107.03342

Lakshminarayanan, B., Pritzel, A., & Blundell, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, *30*.

Pope, C. A., III, & Dockery, D. W. (2006, 6). 2006 critical review - health effects of fine particulate air pollution: lines that connect. *Journal of the Air and Waste Management Association*, *56*(6). Retrieved from `https://www.osti.gov/biblio/20772646`

Siddique, T., & Connor, H. (2022). A survey of uncertainty quantification in machine learning for space weather prediction. *Geosciences*, *12*. doi: https://doi.org/10.3390/geosciences12010027

Valdenegro-Toro, M. (2021). Keras unceartanty. *GitHub repository*. doi: \url{https://github.com/mvaldenegro/keras-uncertainty}

Valdenegro-Toro, M. (2022). Uncertainty quantification in machine learning.
doi: https://mvaldenegro.github.io/files/HZDR-2022-uncertainty-neural-networks-vision-robotics.pdf

Valdenegro-Toro, M., & Saromo, D. (2022). *A deeper look into aleatoric and epistemic uncertainty disentanglement.* arXiv. Retrieved from `https://arxiv.org/abs/2204.09308` doi: 10.48550/ARXIV.2204.09308

Wen, Y., Vicol, P., Ba, J., Tran, D., & Grosse, R. B. (2018). Flipout: Efficient pseudo-independent weight perturbations on mini-batches. *CoRR*, *abs/1803.04386*. Retrieved from `http://arxiv.org/abs/1803.04386`

Wikipedia. (2022). Time series. Retrieved from `https://en.wikipedia.org/wiki/Time_series`