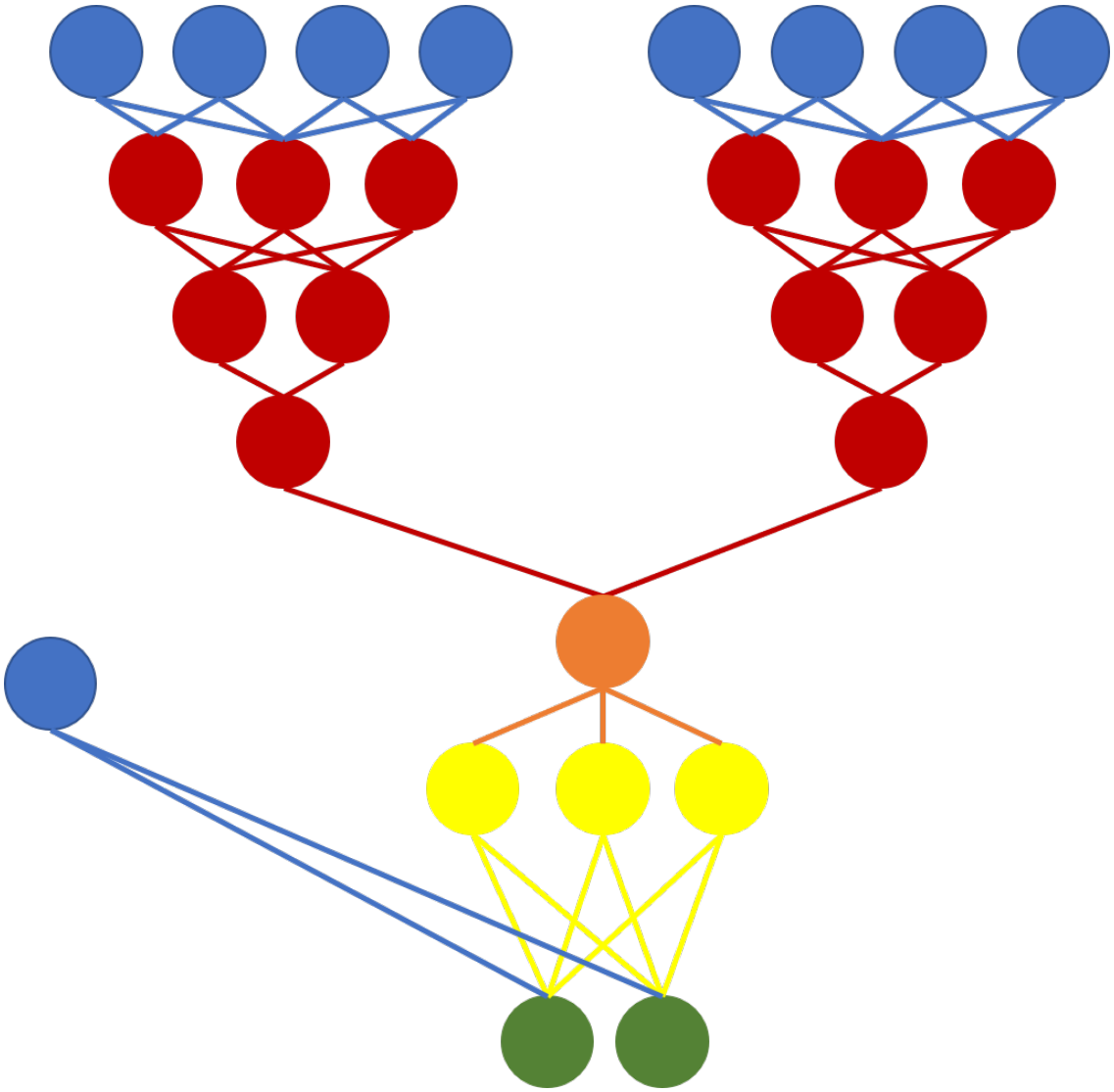
Predicting water levels in the Rhine river using Temporal Convolutional Networks

Rick Dijkstras4137957

july 26th 2019



Contents

Abstract

To ensure that the people in The Netherlands keep dry feet, Rijkswaterstaat (a Dutch governmental institute) wants to know whether deep learning models can be a valuable addition to current water level prediction models. For this, we set as a reference point the multilinear regression model LobithW, which Rijkswaterstaat uses to predict water levels up to two days ahead in Lobith. We developed LobithAI, using Temporal Convolutional Networks. Due to missing data, it was impossible to have a direct comparison between the two models. Instead, we used two different evaluation strategies. When compared on data in the same time range, LobithAI outperforms LobithW on both RMSE and precision. When compared with LobithW directly after its validation, LobithAI scores higher on the RMSE, but LobithW has higher precision for predictions one day ahead. Extensions to LobithAI are also possible, with the best performing extension being the increase of the number of predictions per day. We also inspected LobithAI using activation adjustment, showing that the model mostly uses the closest and most recent measurements. While the comparison between the models was incomplete, the results from the evaluations and the flexibility of LobithAI make it a valuable addition to the models of Rijkswaterstaat.

1 Introduction

Predicting the water level of rivers is of vital importance in The Netherlands: it ensures safety, smooth shipping traffic, and properly executed operational water management. At the moment Rijkswaterstaat (the Dutch governmental institute charged with above named tasks) is using multiple models to predict the water level (e.g. Bergstrom (1995), Dhondia & Stelling (2004), Haag (2012)). However, these models only incorporate hydrological or physical principals or in some cases simple statistical principles.

Current developments in research on smart data analytics has peaked the interest of Rijkswaterstaat. With one of the areas they are interested in being Artificial Intelligence (AI). They wondered whether AI is able to develop a model that has added value to the models they have already incorporated. In order to investigate this I have focused my research on neural networks. And focused on the prediction of river water levels for the Rhine river at Lobith for two days ahead. For this I developed a deep learning neural network called LobithAI. This model is built to compare with a model that Rijkswaterstaat already is using. This model, which is a multilinear regression model, is called LobithW (Haag, 2012).

1.1 Neural networks and water level prediction

The topic of modelling in the hydrological field has been researched extensively. One of the tools used in modelling is neural networks. Neural networks and in particular deep learning neural networks show promising in predicting data that is not very logical at first sight (Marçais & De Dreuzy, 2017), meaning that there is a need to research the added value of deep learning for forecasting water levels.

An extensive review on the usage of neural networks specific for river water level prediction up until 2012 is given in the paper from Abrahart et al. (2012). In this paper however, most examples are a version of a feed forward neural network in the form of a multilayer perceptron. And while there are also some examples using fuzzy neurons or recurrent structures, no deep learning techniques have been mentioned.

Moreover, the papers that do exist in the field of Hydrology concerning deep learning, are not focused on forecasting water levels in rivers. They are rather focused on precipitation forecasting (Tao et al., 2016) or on forecasting ground soil levels (Song et al., 2016).

The closest to forecasting river water levels is the paper from Zhang et al. (2018). In this paper the authors try to combine the input from rain gauges and water level sensors in different neural networks in order to predict water levels in sewers. They found that from a multilayer perceptron, a wavelet neural network, LSTM network and GRU network, the LSTM and GRU network had superior results.

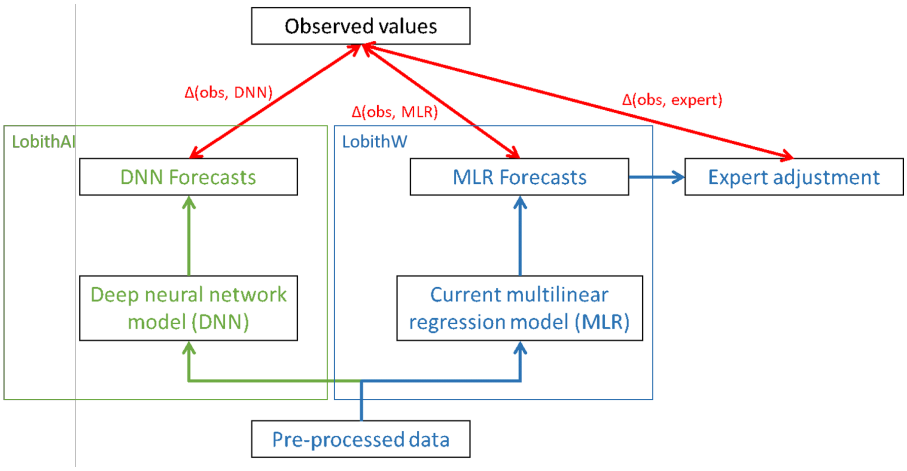


Figure 1: Overview of comparisons

Another paper coming close is researching the prediction of rainfall runoff using LSTMs (Kratzert et al., 2018). Rainfall runoff refers to the amount of rainfall that directly runs off to the river and has a high correlation with the level of the river water. In this study Kratzert et al. found that the LSTM model was able to outperform the benchmark model SAC-SMA + Snow-17 (Newman et al., 2015), which is a technique used in one of the models of Rijkswaterstaat.

1.2 LSTM versus TCN

The dimensionality of the data used to predict water levels is an important factor in deciding which method to use: Some of the data is measured on a two-dimensional service whereas other data is only measured at a few measurement points that are not evenly distributed. Additionally, the data has a temporal dimension.

One candidate to handle temporal data is the Long Short Term Memory architecture (Hochreiter & Schmidhuber, 1997). This architecture has had success in the prediction of precipitation data, which is data used in predicting water levels, in the research by Xingjian et al. (2015).

However, in recent research Bai et al. (2018) showed that a specific type of convolutional neural networks is capable of outperforming LSTMs. This type of network that is called a temporal convolutional network (TCN). The benefits of TCNs over LSTMs is that TCNs appear to be more accurate than LSTMs, while they are also simpler and clearer in use, mainly because of a lack of gated units.

Because of the performance of TCN over LSTM, TCN will be the architecture for LobithAI, the new model.

1.3 Comparing LobithAI with other models

In order to research whether and how much a neural network can improve predicting river water levels a comparison must be made with existing models. Because the models available have different inputs and outputs, a selection had to be made. For this selection we have considered the data available, the complexity of the current model, and the size of the data processed. For these reasons, we have picked the LobithW model to be the current comparison model. Which is a multilinear regression model.

Besides the fact that the LobithW has the smallest scope of the available models and processes the least data, there is another benefit: Ever since LobithW is in production, the predictions made by LobithW have been used by experts, so that these experts make their own adjusted predictions. Using several rules of thumb, such as the presence of snow in certain areas, they create their own predictions, which are rounded on 5 cm.

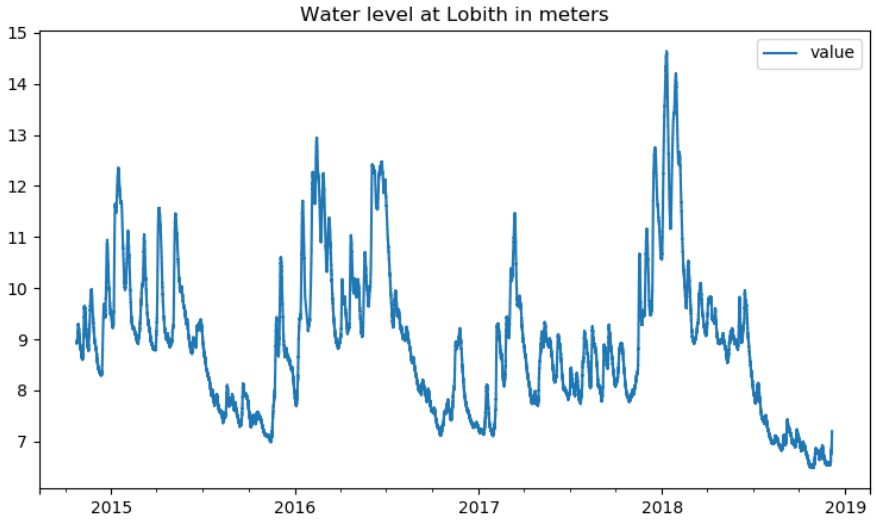


Figure 2: The water level data at Lobith between October 2014 and December 2018.

Combining the LobithW model, the LobithAI model and the expert adjustments, the comparison of the model will be as given in figure 1.

The blue route in this picture is the current situation at Rijkswaterstaat, where the preprocessed data is converted into a prediction by LobithW. These predictions are then adjusted by the experts, which create their own predictions. The green route is the newly added process with the LobithAI model. It uses the same data to create its own predictions. The difference between the predictions and the observed reality will be compared. By having this comparison it is possible to answer the question: Will deep learning techniques be able to reach better predictions than current existing models?

2 Data

For a complete comparison, it is of vital importance to use as much of the same data as possible for the calibration of LobithW and the training of LobithAI. However, in practice it was only possible to have the data used to train LobithAI originate from the same sources as from LobithW. These sources used to calibrate LobithW are t‘he measurement stations in the river Rhine and the weather stations in the surrounding catchment area of the Rhine.

The measurement stations used are stationed at multiple points upstream of Lobith. From the measurements that they make, both LobithW and LobithAI only use the measurement of the water level. The stations used by LobithW are located at Maxau, Plochingen, Worms, Wuerzburg, Kaub, Kalkofen, Trier, Cochem, Koblenz, Andernach, Koeln, Hattingen and Lobith itself. In Lobith the water level is measured and stored every ten minutes in Matroos (a database from Rijkswaterstaat), the measurements from the other stations are stored every thirty minutes. The data used for each prediction from LobithW is based on the measurements 0, 24, 48, 72, and 96 hours in the past. While calibrating LobithW all measurements at all stations are being used. However, the calibrated version of LobithW uses only some moments from some locations (Haag, 2012). An example of the data is given in figure 2

The weather stations used are located in the catchment area from the Rhine river. They are located in Borken, Duesseldorf, Frankfurt, Giessen, Nancy, Strassbourg, Stuttgart and Trier. The stations update their data once every day. For the calibration of LobithW, the observed precipitation from these weather stations was used. Each prediction uses data 0, 24 and 48 hours in the future. However, when LobithW is used in production, there is no observed precipitation

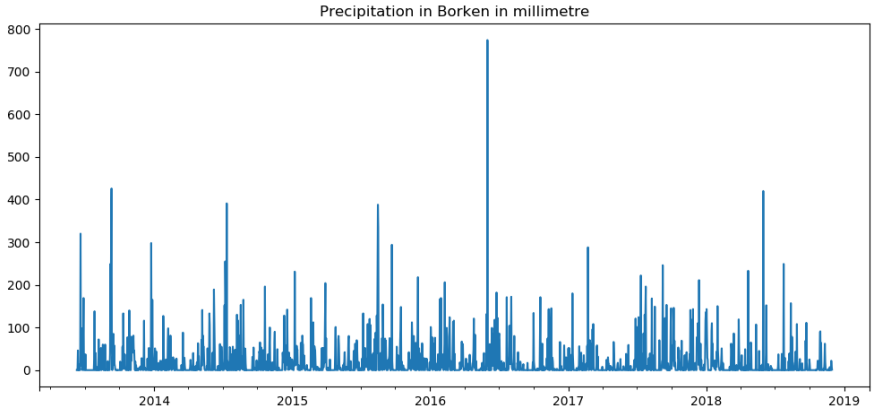


Figure 3: Precipitation data from the Borken weather station between April 2014 and December 2018

for the future available, so predicted precipitation is used. In order to keep LobithAI as close to LobithW as possible, and because the predicted precipitation data from the past years was unavailable, we trained LobithAI on observed precipitation as well. An example of the observed precipitation is given by figure 3.

The data used to train LobithAI ranges from the period of 2010 to 2018. The data used to calibrate LobithW ranges from the years 1982 to 2011. This results in a difference in the data used for training LobithAI and calibrating LobithW.

Because of this difference it is important to evaluate in two different ways. On the one hand we compare the results of both models on the time frame that they are trained on. On the other hand, we can compare the results of LobithW in production for the data that LobithAI is trained on. For this latter period LobithW uses predicted precipitation where LobithAI is using observed precipitation. This gives a disadvantage to LobithW in its performance. And while it is possible to compare LobithAI to the experts in the same period, they are also using predicted precipitation. More details about the evaluation are available in section 4

3 Methods

3.1 LobithW

LobithW is a multilinear regression model. This is a statistical model consisting of different multilinear regression equations. Such an equation is given by:

$$y_n = \beta_0 + \beta_1 \times x_1 + \beta_2 \times x_2 + ... + \beta_i \times x_i$$

The dependent variable y_n is the water level that is calculated. The explanatory variables x_1 through x_i are either the water level at different measurement stations in the Rhine river or precipitation levels. β_0 is the constant term and β_1 through β_i are the regression weights for each explanatory variable.

The regression equations from LobithW are split up in different sub models depending on certain scenarios. These scenarios are the water level in Lobith being above or below 10 meters, or the discharge of the river having a value above 2300 m^3/s or for extreme high water above 5000 m^3/s. Which sub model is used at which time is dependent on the water level and discharge in Lobith. Besides these scenarios, the predictions for one day and two days ahead both have their own equations. From the available sub models from LobithW, three of the models are still being used today. These are the models for a water level below and above 10 meter and the discharge above 5000 m^3/s.

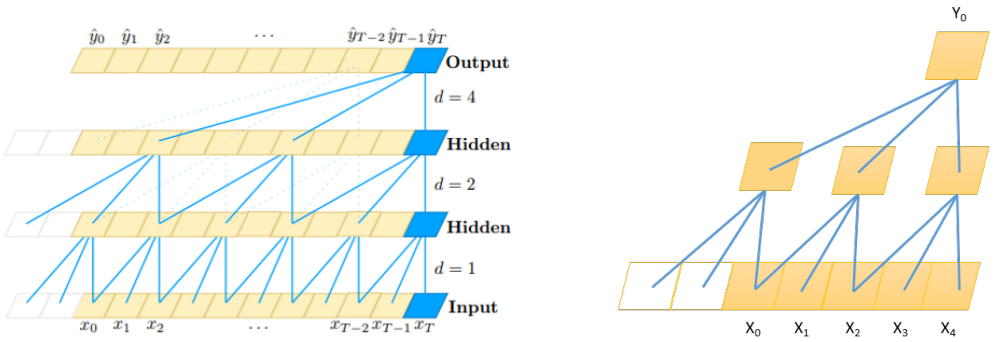


Figure 4: a) On the left an example of the regular TCN network with two hidden layers and an increasing dilation, as given by Bai et al. (2018). b) On the right an example of a one-dimensional TCN network similar to the TCN network used in LobithAI. This network has a stride of two.

There is a difference in some of the data used in the calibration of LobithW and the data being used now: in the calibration phase, LobithW was trained using observed precipitation. Currently in production, it uses predicted precipitation. This predicted precipitation is less accurate and has a lower variance than the observed precipitation, which probably has an influence on the performance of LobithW.

3.2 Temporal Convolutional Network

A temporal convolutional network (TCN) as used by Bai et al. (2018) is a sequence-to-sequence mapping. This mapping is from a certain input sequence $X_0, ..., X_T$ to a corresponding prediction output sequence $Y_0, ..., Y_T$, with the constraint that for the prediction of Y_t, the only available input is $X_0, ..., X_t$. Meaning that future information from $X_{t+1}, .., X_T$ will not be used.

In order to achieve this, the TCN is based on a fully convolutional network, with as difference that it uses causal convolutions. Meaning that there is no leakage from future time points to the past. In order to map between two sequences of the same length, a TCN uses zero-padding at the beginning of the input sequence in order to keep the layers the same length. In order to decrease the numbers of layers needed to cover more data, the model additionally uses increasing dilation rates between the input and the output, which ensures that input data is not processed to many times. The resulting network is shown in figure 4 a.

The TCN network used in LobithAI is different from the regular TCN network: instead of mapping one input sequence to one output sequence of the same length, the output has a different length in the temporal axis from the input. And the input is more than one input sequence. This means that there need to be a few changes to the TCN network.

Firstly, LobithAI is not keeping the same size per layer, but instead is working to converge to a layer with only one feature map. While it is still necessary to create some zero-padding in situations where the kernel size and stride combined need a larger input to get to the correct size of output. Secondly, instead of using dilations, LobithAI uses stride. With the same input size, dilation creates a larger output with a smaller kernel sizes than using stride. In order to reach an output of length 1, a network using stride uses the same input as a network with dilation, but is smaller due to the lack of neurons in hidden layers that will not be used for the output. The resulting one-dimensional network for LobithAI is shown in figure 4 b. Finally, the TCN network in LobithAI has an extra dimension because of the multiple input sources from the different measurement stations. As a result instead of using a one-dimensional fully connected convolutional network, a two-dimensional network is created. Where we still use zero-padding on one side per dimension to ensure a fit between input size and stride to the output size.

3.3 Different inputs

The input for the models is given by two distinct sources: the water level data and the precipitation data. These data sources differ in a few aspects: The water level is given in meters, spanning four days in the past, is the value measured at one time point and has no fixed limits, whereas the precipitation data is given in millimetres, spanning two days in the future, where the value is the sum over 24 hours and has a lower limit at zero millimetre.

With these difference in mind, both the precipitation data and the water level data are processed by their own TCN network, until the temporal and spatial dimensions have a length of 1. When this point is reached, the resulting feature maps from both networks are combined and further processed in one dense network per prediction day until a predicting value is reached. By having one dense network per output, it is possible to have a flexible output length.

When the predicting value is a relative value, the network will make smaller errors in prediction. This is done by adding the current water level to the predicting value. This way, the absolute height of the water is added and only the relative change is needed to calculate the correct prediction. The resulting architecture is shown in figure 5.

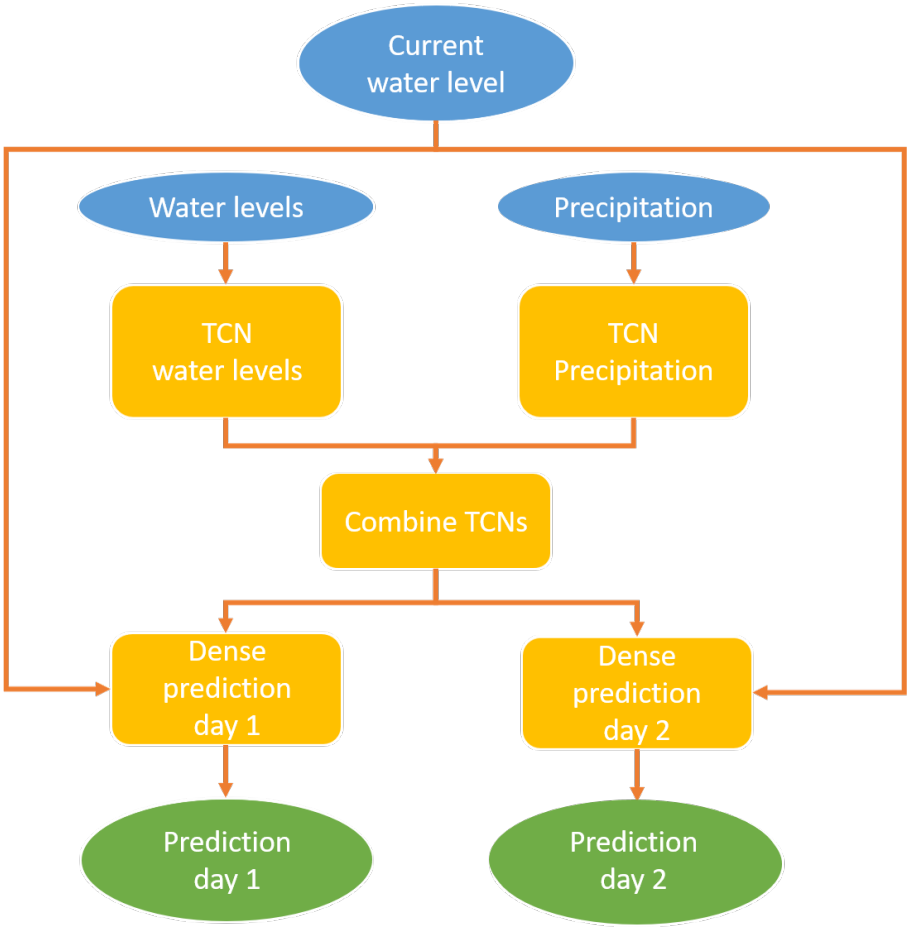


Figure 5: Basic architecture from LobithAI.

3.4 Splitting training and validation

LobithAI is trained using 25-fold cross-validation. However, contrary to normal k-fold cross-validation, the data is split using temporal blocking Bergmeir & Benítez (2012). This means that the data is not split completely random in a training or a validation set, but instead that blocks of

roughly four months have been made. The reason for this length is that the water level in Lobith has a yearly seasonal effect, and even a half yearly effect with the water rising. This can be seen in the autocorrelation plot in figure 6. To ensure that the validation blocks do not have repeating patterns from seasonal effects, the blocks have been made smaller than 6 months, without getting to small lengths, which results in to many training sessions.

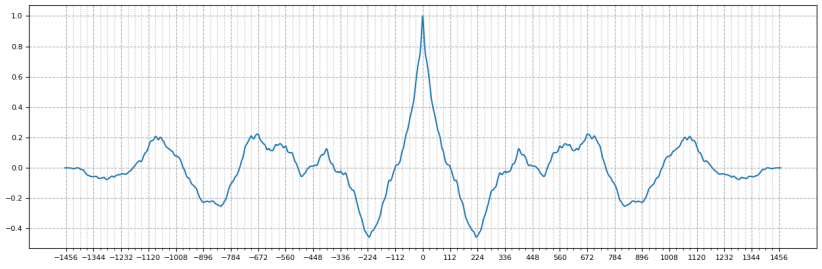


Figure 6: Autocorrelation for the water level data from Lobith from 2014 to 2018.

3.5 Implementation details

LobithAI is implemented in the Python programming language using the Keras library (Chollet et al., 2015). The loss function for the training is the Root Mean Squared Error, which matches one of the evaluation criteria. The activation function of the network is the ReLu activation (Krizhevsky et al., 2012). The kernels are initialised using he normal initialisation (He et al., 2015). The optimiser is the RMSprop algorithm (Tieleman & Hinton, 2012) with a learning rate of 0.001, $\rho = 0.9$ and $\epsilon = 10^{-6}$.

At the start of the model, the receptive field size for the water level data is 3 by 5 for the temporal and spatial dimension respectively and the receptive field size for the precipitation data is 3 by 3 for the temporal and spatial dimension respectively. The stride used for the water level data is 2 by 4 for the temporal and spatial dimension respectively, whereas no stride is used for the precipitation data. The training of the network stopped after either 400 epochs or when the RMSE of the validation did not increase for 50 epochs.

4 Evaluation

As mentioned in section 2, the data used to calibrate LobithW was unavailable to train LobithAI. Because of this, different training data is used to train LobithAI. This difference in data made a direct comparison between the performance of LobithW and LobithAI impossible, and caused two different manners of evaluating.

The first method of evaluating is by comparing the models based on the evaluation after their validation. LobithW is evaluated on the data between january 1st 2006 and december 31st 2011. LobithAI will be evaluated over the data between may 13th 2010 and december 11th 2018. With this evaluation the data used will be from the same sources, but different time ranges.

The second method of evaluating is by comparing the models over the same time range. Which means that we use the results from LobithW in production and additionally the adjusted predictions from the experts. The data used is from februari 16th 2014 to december 11th 2018. The problem here is that both LobithW and the experts had no access to observed precipitation data, which means that the data used to compare is from different sources.

One way to fix the problem of different data in the second method would be to have the LobithW model create predictions based on the data that is now available. However, it was impossible to recreate the LobithW model, because of missing settings in the model in the paper from Haag (2012).

Two criteria are used to evaluate: The root mean squared error (RMSE) and the percentage of predictions within given boundaries (percentage within bounds). These two criteria have been used in evaluating LobithW as well.

4.1 RMSE

The first criterion for evaluation is the root mean squared error. This error is used commonly in the field of Hydrology (e.g. Boyle et al. (2000), Henriksen et al. (2003)). Additionally it is available as a loss-function for neural networks, which makes it possible to train LobithAI on the RMSE. The formula for RMSE is given by:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{n}(obs_i - prd_i)^2}$$

Where the N stands for the number of predictions, prd_i for the predicted values and obs_i for the observed values. Using the RMSE, larger deviations from the observed values are punished more than small deviations. This results in models that have less extreme outliers to have lower and thus better RMSEs. For the calculation of the RMSE, obs_i and prd_i are in centimetre.

4.1.1 RMSE during validation

The performance of LobithW after its validation is reported by Haag (2012) for four different sub models. These sub models are either divided by the amount of water flowing through the river (discharge) or by the water levels: There is a model for water levels (H) at Lobith above and below 10 meters and for a discharge (Q) above 2300 and above 5000 m^3/s. Tabel 1 shows the RMSE's for all different models.

	LobithW				LobithAI
Day	H<10	H≥10	Q≥2300	Q≥5000	
1	4.62	5.4	5.7	10.5	**4.09**
2	**8.2**	10.0	9.5	12.7	9.19

Table 1: RMSE for the validation for the sub models from LobithW as well as for LobithAI. Bold values show the best values.

When evaluated on the RMSE, LobithAI is the best model for predictions one day ahead with a RMSE of 4.09. However, for a prediction two days ahead the best results are obtained by the sub model from LobithW for water levels under 10 meters, with an RMSE of 8.2. These better results for the In order to further compare the working of LobithAI it was also split in in observed waterlevels above and below 10 meters.

By splitting the results from LobithAI between water levels below 10 meter and above 10 meters, it showed that below 10 meters the RMSE for LobithAI is 6.06. Which is an improvement in comparison to the 8.2 from LobithW. For water levels above 10 meters however, LobithAI had an RMSE of 10.0. Which is the same for the sub model from LobithW. Following this results, LobithAI is either equal to or better than LobithW compared to their calibration state.

4.1.2 RMSE for same the time period

The results from LobithW that uses the same time period as the data where LobithAI is validated on, are given by the predictions that the model gives in production. Additionally, the predictions from experts that adjust the results from LobithW are also available for this time period. Instead of using all sub models of LobithW for all dates, only the sub model created for the particular situation is used. Because the prediction from LobithW and the experts are created in production, observed precipitation from two days later is not available, and predicted precipitation, which has less accuracy is used. The results are shown per time period for which there is a validation split

from LobithAI. The results for predictions one day ahead are shown in figure 7. Results for two days ahead are shown in figure 8.

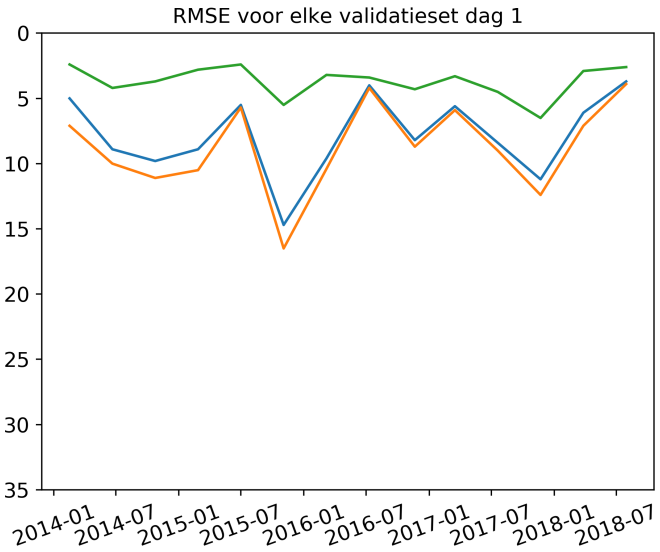


Figure 7: RMSE for predictions one day ahead. Shown per validation split from LobithAI between 2014 and 2018. The blue line is the RMSE from the experts, the orange line is the RMSE from LobithW and the green line is the RMSE from LobithAI.

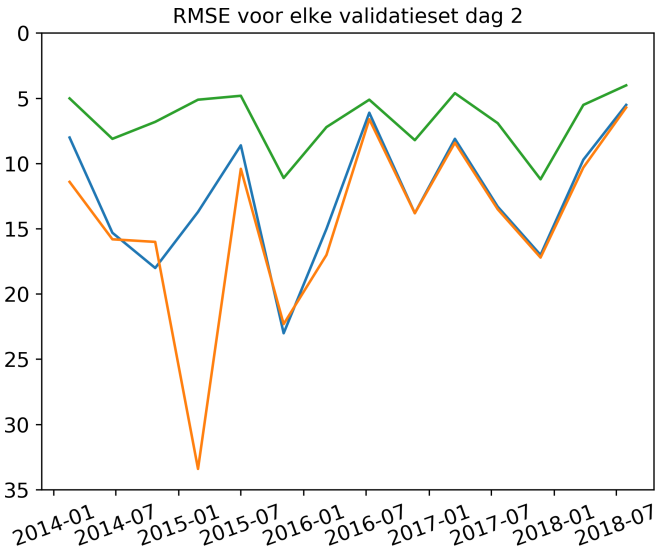


Figure 8: RMSE for predictions two days ahead. Shown per validation split from LobithAI between 2014 and 2018. The blue line is the RMSE from the experts, the orange line is the RMSE from LobithW and the green line is the RMSE from LobithAI.

Both figures 7 and 8 show that there is hardly any difference between the RMSE for the experts and LobithW. For one day ahead the experts do show slightly better results. Two days ahead however, shows LobithW sometimes having better results than the experts.

LobithAI on the other hand is always outperforming both LobithW and the experts by having a lower RMSE. Still at the same periods that LobithW and the experts have a relatively higher RMSE, LobithAI also has a relatively higher RMSE.

4.2 Percentage within bounds

Upon the creation of the LobithW model, certain demands where made by the users of the model about how accurate the model should be. Those demands are set for up to four days ahead. They can differ depending on the water level being higher or lower than ten meters. The values of these bounds are given in table 2. For the evaluation between LobithAI and LobithW, we only use the values for up to two days ahead, since the predictions are only up to two days ahead.

Days ahead	low water level	high water level
1	5 cm	10 cm
2	15 cm	15 cm
3	20 cm	20 cm
4	25 cm	40 cm

Table 2: Desired accuracy of water level in cm

4.2.1 Percentage within bounds during validation

Using the bounds set for LobithW, the percentage of predictions falling within these bounds has been calculated. This has happened for the four sub models from LobithW for water levels (H) below and above 10 meter and discharges (Q) above 2300 and 5000 m^3/s. For the sub model for water levels below 10 meter, the percentage within bounds was a lot lower for the prediction one day ahead than for the prediction two days ahead. According to Haag (2012), this difference was probably due to the stricter bounds for water levels under ten meters one day ahead.

The percentages from the sub models from LobithW as well as the percentage from LobithAI are given in table 3. This table shows that for two days ahead LobithAI is the most accurate model with a percentage of 94.4%. For one day ahead however, the LobithW sub model for water levels above ten meters achieves better results with a percentage of 94%. Even after splitting the predictions from LobithAI in water levels above and below 10 meter, the accuracy for LobithAI for water levels above 10 meters does reach only 90.6%.

So while LobithAI is better two days ahead, it still achieves results lower than LobithWs sub model for water levels above ten meter. An interesting side-note here is that this sub model is not used in production and is instead replaced with the sub models for discharges, because there was no significant difference found between them (Haag, 2012).

	LobithW				LobithAI
Day	H<10	H≥10	Q≥2300	Q≥5000	
1	82%	**94%**	90%	87%	89.1%
2	92.5%	88.5%	88%	93.5%	**94.4%**

Table 3: Percentage within bounds for LobithW (split in sub models) and LobithAI. Bold values show the best values.

4.2.2 Percentage within bounds for the same time period

Similar to the RMSE, the percentage within bounds is also compared on data in the same time frame. These results are given in figure 9 for predictions one day ahead and in figure 10 for predictions two days ahead.

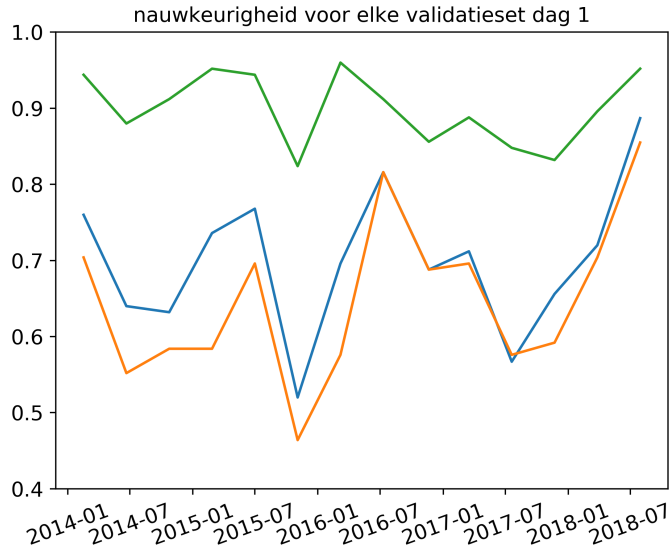


Figure 9: Percentage within bounds for predictions one day ahead. Shown for validation splits from LobithAI from 2014 to 2018. The blue line is the RMSE from the experts, the orange line is the RMSE from LobithW and the green line is the RMSE from LobithAI.

For the predictions one day ahead we see that there is a sleight difference in percentage within bounds between LobithW and the experts. LobithAI however, has a better percentage within bounds for every period than both LobithW and the experts.

Still all models seem to have trouble at similar parts of the data. With the worst performance being the period at the end of 2015 and the best period halfway 2018. This last result was probably due to the low and stable water level of the Rhine river halfway 2018.

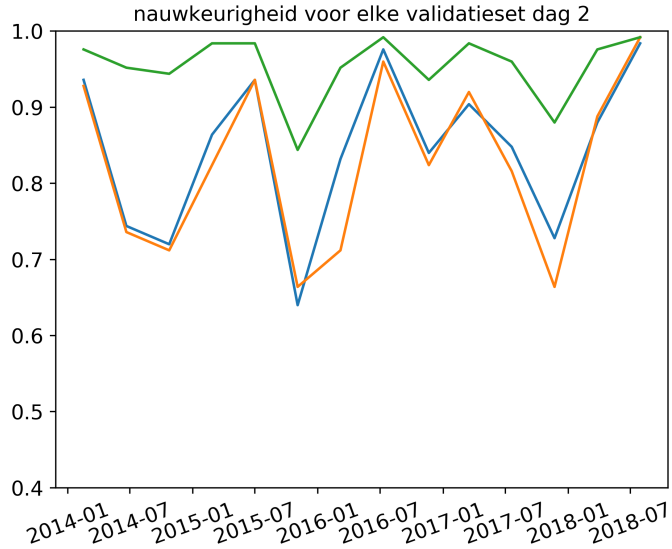


Figure 10: Percentage within bounds for predictions two days ahead. Shown for validation splits from LobithAI from 2014 to 2018. The blue line is the RMSE from the experts, the orange line is the RMSE from LobithW and the green line is the RMSE from LobithAI.

Figure 10 shows the percentage within bounds for the predictions two days ahead. When compared to the predictions one day ahead, the precision has improved a lot. For some periods LobithAI even has a precision of 1. Meaning that for a period of 4 months all predictions fell within the given bounds. The reason of the improvement for the second day is probably the more relaxed bounds for errors two days ahead in comparison to the bounds for one day ahead. An interesting finding here is that LobithW seems to have better results than the experts, while the experts are trying to improve on the results of LobithW.

4.3 Rounding

When comparing LobithAI to the predictions of the experts, another factor has to be taken into account: The predictions from the experts are all rounded to the closest multiple of 5 cm. In order to examine whether LobithAI is benefiting from having a higher prediction precision, the predictions from LobithAI were rounded to 5 cm. This had some minor changes for the RMSE and the percentage within bounds:

The RMSE slightly worsened for all validation periods, with a maximum of at most 0.5. The new RMSE from LobithAI is closest the experts with a difference of only 0.2 for the predictions one day ahead in the period from July to November 2016. However for all other validation sets, the difference in RMSE was at least 2.0.

The percentage within bounds sometimes improved and sometimes worsened because of the rounding. However, the smallest difference was 5%, with LobithAI still performing better than LobithW or the experts.

5 Model analysis

LobithAI is a deep neural network. One of the properties of neural networks is that they work like a black box. Making it unclear why a network makes a certain decision, what a decision is based on, why there certain mistakes are made and what kind of input will definitely result in errors.

To start unravelling the black box, some techniques have been developed. These techniques are mostly developed for neural networks that are tasked with classification (e.g. Erhan et al. (2009), Simonyan et al. (2013)). However, these techniques can be translated to regression networks. One of these techniques is used to inspect a trained version of LobithAI.

5.1 Adjusting activation

One of the techniques to understand classification neural networks, is called activation maximisation (Erhan et al., 2009). This is done by checking what input activates a class category the most.

LobithAI however, does not classify into categories, but instead is a regression network. So, while it is impossible to see the highest activation of a class in LobithAI, it is possible to see what the effect is on the predicted water level by adjusting the activation. This is realised per value for the input of the water level for all five days at all 13 locations.

Because the network is trained on normalised values, it is possible to give the network an input where all values are 0, except for one of the inputs. That one input has a value of either 1 or -1 when the activation is adjusted respectively upward or downward. The resulting predicted value is then shown in a heatmap at the location of the adjusted input. This results in the heatmaps as shown in figure 11. The figure shows that mostly changes in the most recent days have an influence on the predicted value.

For the predictions one day ahead, there is only a marginal effect from the input adjustments at locations past Koblenz. For two days ahead, the data for locations until Kaub might make a difference but location further upstream do not. This difference of effect for prediction days ahead between Koblenz en Kaub for one day ahead, can be explained by the fact that it takes roughly a day for the water from the Rhine to travel from Kaub to Koblenz. However, from Koblenz, it

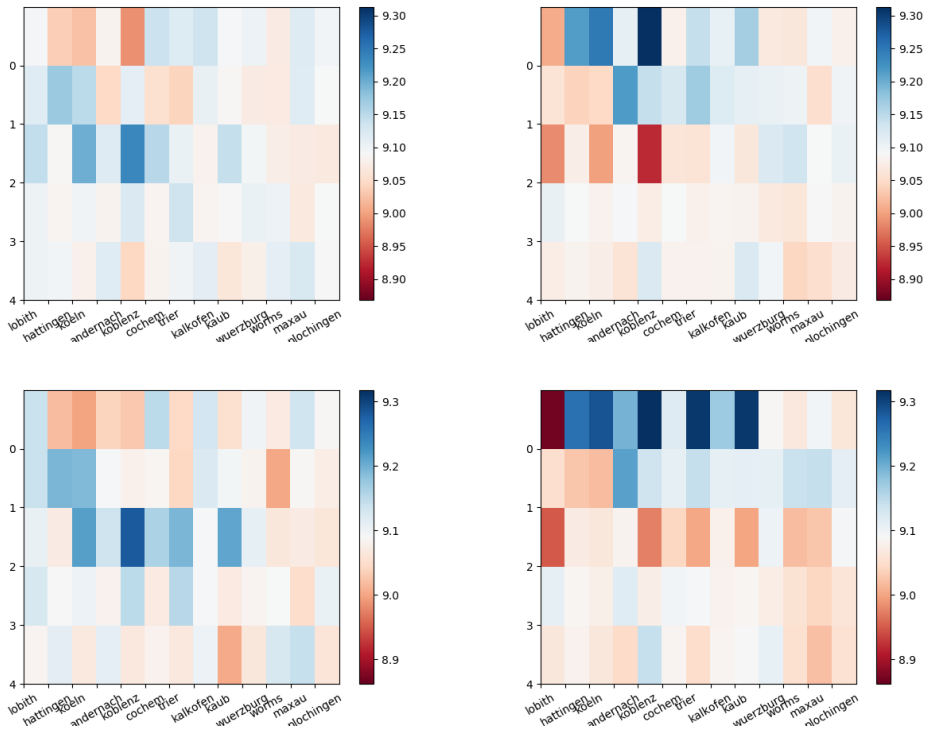


Figure 11: Heatmaps showing the effect of raising single values on the output. On the left are the effects of lowering the inputs, on the right the effects of raising the inputs. The heatmaps above represent the predictions one day ahead, those below the predictions of two days ahead.

still takes another two days to reach Lobith. Apparently, the model is responding to a relative increase between day two and day three.

This response to relative increases is also showing in the difference between the effect of data zero days in the past from Koblenz and data 2 days in the past from Koblenz. The data zero days in the past is enforcing the effect of having either more or less water level as input, whereas data from 2 days ago is giving rise to a reversing effect. This reversing effect can be explained by seeing it as the model seeing a huge rise in water level in Koblenz in two days time, meaning that it will probably keep rising.

A drawback from the activation adjustment method is that we only look at individual values. However, LobithAI seems to be reacting on relative differences between values, such as the relative difference between day 0 and day 2. New research will have to look into the possibility of including relative effects.

Still, this method shows that LobithAI mainly focusses on water levels at Lobith, Andernach, Koblenz and Kaub. With this information we know that measurement errors occurring in one of these locations will give rise to incorrect values.

6 Extending LobithAI

One of the benefits of LobithAI in relation to LobithW, is that its basic architecture is easily extendable. Some of these extensions we have already examined. For the results of this extensions, one has to keep in mind that the results might be slightly worse than the results from LobithAI, because the current network has been optimised for the current architecture. Still, even with this drawback, it is possible to research the effects of changing the network.

Dag	LobithAI	+ moments in	+ stations in	+ days in	+ days out	per hour out
1	4.8	6.4	6.0	5.8	8.1	**4.3**
2	9.1	16.2	8.5	8.4	11.3	**7.2**
3	-	-	-	-	21.5	-
4	-	-	-	-	37.8	-
5	-	-	-	-	57.2	-
6	-	-	-	-	76.7	-
7	-	-	-	-	94.5	-
8	-	-	-	-	110.0	-

Table 5: RMSE for the extensions of LobithAI.

The extensions made are: increasing the amount of measurements per day, increasing the locations of measuring, and increasing the number of days both in the input and in the output. Additionally, the model has been evaluated on predicting more than two days into the future.

In order to decrease the time needed per experiment, only one data set has been used as validation set. The validation set chosen for the evaluation is chosen as a representative moment of the data. It is the data between October 20th 2017 and February 18th 2018. This period has an irregular pattern for the water levels with levels between 7.7 and 14.6 meter. Results for LobithAI in that period are shown in table 4. Results for the extensions are shown in table 5 and 6.

Day	RMSE	Percentage within bounds
1	4.8	88.6%
2	9.1	91.1%

Table 4: Results from LobithAI on the validation period between November 2017 and March 2018.

6.1 Increased number of measurements per day

The LobithW sub models use as input for the water levels the data from the past 5 days at multiple measurement stations in the Rhine river. From these measurements only one measurement per day is used. However, the measurement stations in the Rhine river have an update frequency of once per half hour. This enables the possibility to increase the amount of measurement moments up to 48 measurements per day.

In the experiment for increasing the moments of measurements, the frequency is updated to once per hour. The results show that increasing the amount of inputs, results in an increase of RMSE (+1.6 and +7.1) and a decrease of precision (-4.1% and -11.4%). One possible reason for the worse results is the fluctuation in the water level from hour to hour. In the course of a day, this fluctuations have been averaged out, resulting in more stable predictions.

6.2 Increased number of measurement stations

Besides raising the amount of measurement moments in a day, it is also possible to increase the amount of stations used. LobithW only uses 13 of the 27 available measurement stations located in the Rhine river. For the experiment in increased number of measurement stations, all available stations have been used.

When making predictions one day ahead, this increase of measurement stations result in worse RMSE (+1.2) and percentage within bounds (-1.4%). However, for predictions two days ahead there is an improvement in both RMSE (-0.6) and precision (+0.8%). So, where one day ahead, the current measurement stations are enough, the amount of stations for two days and maybe even more days ahead, will improve from using more measurement stations. A possible reason for the improvement on the second day is that information from tributaries, such as the Wieb river, are added to the model.

Dag	LobithAI	+ moments in	+ stations in	+ days in	+ days out	per hour out
1	88.6%	83.7%	86.2%	83.7%	74%	**90.2%**
2	91.1%	79.7%	91.9%	91.1%	80.5%	**95.1%**
3	-	-	-	-	74%	-
4	-	-	-	-	61%	-
5	-	-	-	-	54.5%	-
6	-	-	-	-	54.5%	-
7	-	-	-	-	48.8%	-
8	-	-	-	-	50.4%	-

Table 6: Percentage within bounds for extensions of LobithAI.

6.3 Increased number of input days

Besides increasing the measurement moments per day, it is also possible to increase the amount of days that we look back in time for the water levels or look forward or even backward for the precipitation data. For the latter we have to be careful in how far we will look ahead, since predicted precipitation will lose accuracy upon increasing the range we want to predict. For this reason, the extension was made with looking back in the past for 10 days with both water level and precipitation and looking in the future for only two days for the precipitation data (which is the standard situation for LobithAI).

Adding an increased number of input days seems to have a positive effect on the RMSE on the second day, since it decreases. However, for the predictions one day ahead both RMSE and precision seem to perform worse and the precision for one day ahead stays the same. The reason for the decreased performance for the first day might be due to the extra data not giving extra information and instead giving rise to overfitting.

6.4 Increased number of output days

It is possible to extend LobithAI to make predictions for more than two days ahead. While we do not expect the model to improve in performance upon looking ahead for multiple days, it will give some information on how well the current model can predict for more days ahead. For this reason, we have extended the output days from two to eight.

The percentage within bounds for more days ahead is adjusted by giving the model 5 cm of error extra per day, making its resemblance close to the original bounds for precision (only for high water four days ahead is the original more lenient). The precision of the extended LobithAI drops for the first five days, (with an already observed exception for the first day.) after these five days, the precision is stable around 50%. One possible reason for this is given by the fact that the water from the Rhine takes five days from the furthest measurement station in Maxau to Lobith. Predictions for longer than five days cannot rely on current water levels.

Additionally, the extended version of LobithAI has worse performance on precision (-12.6% and -10.6%) and RMSE (+3.3 and +2.2) than the normal version of LobithAI. The reason for this is probably the fact that the first two days of the model have to share their input with more days ahead.

6.5 Predicting more than once per day.

Besides increasing the amount of days that will be predicted, it is also possible to increase the frequency of predictions made per day. In this extension, the predictions from LobithAI are increased from once per day to once per hour.

For this extension we adjusted the evaluation of the percentage within bounds. We set predictions up to 24 hours ahead to fall within the bounds of one day ahead and the predictions from 24 to 48 hours are to fall within the boundaries of two days ahead. In order to compare to the other models, only the values for 24 and 48 hours ahead are given for tables 5 and 6.

Extending the model to predict once per hour instead of per day causes the biggest increase of performance of all extensions. Figure 12 shows the average RMSE and the percentage within bounds per hour for the extended LobithAI. The RMSE is increasing almost linear, which is something that you would expect, since the prediction is more and more uncertain. The percentage within bounds is stable for a few hours, after which it starts to decrease linearly as well and jumping up again at the 24 hour point, where the boundary is relaxed. The precision is fluctuating a little bit more than the RMSE, which is probably due to predictions being close to the allowed boundary, sometimes falling on the right side of the boundaries.

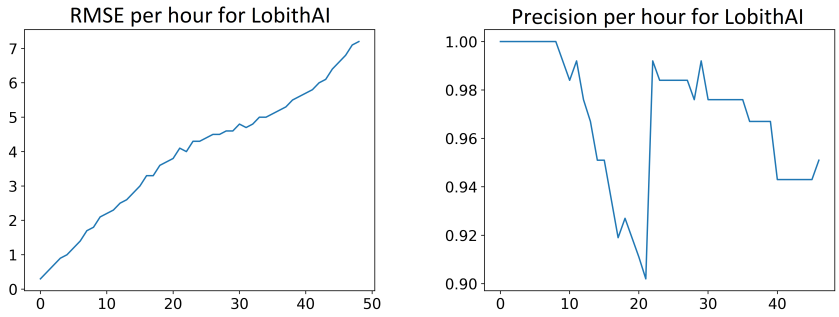


Figure 12: RMSE and percentage within bounds for the extended version of LobithAI with hourly predictions.

7 Discussion

7.1 Possible improvements

LobithAI is one of many possible approaches to creating a neural network that processes the available data and generates predictions. Even with the current architecture there are many different changes possible which might improve the performance of the network.

First of all, a big improvement will be made when the training of the model will be done with the data with which the model will be running in the end. In this case this means that it is important to have access to predicted precipitation data to train the model. Using this data also makes a better comparison between LobithAI and currently used models possible.

Additionally, the data for precipitation and water level measurement stations is 1-dimensionally processed by the network (2-dimensionally if you include the temporal dimension). In reality, the measurement stations are located in a 3-dimensional space, which can be included in the input. Increasing the dimensionality also helps with predicted precipitation data, which is already given in a 2d-grid format.

There are also sources of data available that have not been used yet. For instance some precipitation measurement stations also measure the height of the snow if present and the temperature. The amount of snow and height of the temperature is known to have an effect for more than two days in the future in the form of snow melt (Newman et al., 2015). Adding this information, might also benefit predictions two days ahead.

Furthermore, many types of architecture have not been investigated in this project. For one, an LSTM based network has not been created. This decision was made based on the work from Bai et al. (2018), which showed that convolution networks can create equal or better results on tests where LSTMs traditionally were deemed the best choice. Besides LSTMs, different architectures are also possible. Even the combination of LSTMs with convolutional networks is possible. Since the comparison between LSTMs and convolution networks have been made on different problems than predicting water levels, the LSTM architecture also might add some benefit here.

Another possible improvement is to use LobithAI not as a predicting network, but as an error correcting network. This might be done by adding the predictions made by LobithW to the input

from LobithAI or even the input from the experts. Using this method, LobithAI might be able to increase the accuracy and precision from the predictions from LobithW and the experts.

There is the possibility of adding a dropout method (Srivastava et al., 2014) as well. Using dropout, random weights are disabled during training. This makes it harder for the model to converge on the training set and thus decreases the amount of overfitting. While normal dropout did not improve the results, different versions of dropout, through max pooling or dropping entire layers might add certain benefits (Wu & Gu, 2015).

Finally, the training data can be augmented. With augmentation, the training data will be adjusted slightly from its real values, with random increases and decreases. Doing this might both reduce overfitting and make the model more stable against measurement errors.

Besides these adjustments, there are many more adjustments possible that did not fit within the scope of this research. It is likely that some of the adjustments will increase the performance of the current version of LobithAI.

It is however important to note that some of the adjustments will mostly improve the predictions on predictions for the near future, whereas other adjustment, such as the data for snow melt will impact predictions on a longer scale.

7.2 Conclusion

We developed LobithAI as a temporal convolutional neural network that generates better predictions over all than the current predictions from both LobithW, the current model used, and the experts adjusting the predictions from LobithW. However, whereas LobithAI is better than LobithW and experts in general, this does not mean that it always produces better results. This is especially the case when the water levels and precipitation prediction have values that have not been seen before by LobithAI.

For LobithW, there is a difference in performance between when the model was calibrated and now that the model is in production. One of these reasons is that LobithW is calibrated on data until 2011, where currently there have been changes in the environment producing the water levels. However, because the predictions produced by LobithW still have enough percentage within the bounds set, there will not be a new calibration soon.

The new calibration of LobithW will probably have predictions closer to LobithAI, since this was the case with the last calibration as well. If the resulting increased performance of predictions will be equal to the performance in 2011, this will result in LobithW having better percentage within bounds than LobithAI for predictions one day ahead for water levels above 10 meters. However, the percentage within bounds for two days ahead and the RMSE will still have better values for LobithAI.

An advantage for LobithAI over LobithW is that LobithAI can easily be retrained and even be extended. The retraining will make sure that the performance of LobithAI will continue to be stable and not decline due to small changes in the environment.

Extension of LobithAI such as increasing the days, locations and frequency of input or the days and frequency of the predictions have already been tested in section 6, showing that some of the extensions might really improve predictions, such as increasing the frequency of the output.

For this research it was not possible to use the same data as was used to calibrate LobithW. Instead a comparison was made with the results of LobithW created at the moments for which data was available. This results in a few concerns. First of all, since LobithW was calibrated there have been some changes to the flow area of the Rhine river: the ground underneath the Rhine river sinks a few centimetres every year and recently a lot of the flow area of the Rhine has been changed manually with the "Ruimte voor de rivieren" policy. These changes might be a cause for the worsening performance of LobithW. Secondly, LobithW was calibrated with observed precipitation but is now using predicted precipitation to create predictions.

Because of the missing data, it is impossible to directly compare LobithAI with LobithW: both the water level data used to calibrate LobithW, the predicted precipitation data, with which LobithW is working since 2011 is deleted after 30 days and the details needed to completely recreate LobithW are missing.

Nevertheless, it is still possible to show the contribution of deep learning models to the current models of Rijkswaterstaat. Since LobithAI shows to perform at least equal to LobithW in calibration. Furthermore it can create a prediction in a matter of seconds and retrain the model in 20 minutes on a CPU with 2.7ghz and no GPU. By implementing a daily retraining of the model, it is possible to keep the model up-to-date for a very long time. Additionally, LobithAI is easily extended to improve its performance and give more detailed predictions. When extended to train with predicted precipitation it will probably even increase in performance due to the increased amount of information in the predicted precipitation.

In conclusion, with the possible extensions and modifications possible for LobithAI, the optimisation that is possible when data will be saved from now on, and with the performance of LobithAI being at least equal to the current models and experts adjustments, it can be safely said that AI models will be a valuable addition to the models used by Rijkswaterstaat.

References

Abrahart, R. J., Anctil, F., Coulibaly, P., Dawson, C. W., Mount, N. J., See, L. M., ... Wilby, R. L. (2012). Two decades of anarchy? emerging themes and outstanding challenges for neural network river forecasting. *Progress in Physical Geography: Earth and Environment*, *36*(4), 480-513. Retrieved from `https://doi.org/10.1177/0309133312444943` doi: 10.1177/0309133312444943

Bai, S., Kolter, J. Z., & Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*.

Bergmeir, C., & Benítez, J. M. (2012). On the use of cross-validation for time series predictor evaluation. *Information Sciences*, *191*, 192–213.

Bergstrom, S. (1995). The hbv model. *Computer models of watershed hydrology*.

Boyle, D. P., Gupta, H. V., & Sorooshian, S. (2000). Toward improved calibration of hydrologic models: Combining the strengths of manual and automatic methods. *Water Resources Research*, *36*(12), 3663–3674.

Chollet, F., et al. (2015). *Keras.* `https://keras.io`.

Dhondia, J. F., & Stelling, G. S. (2004). Sobek one dimensional–two dimensional integrated hydraulic model for flood simulation–its capabilities and features explained. In *Hydroinformatics: (in 2 volumes, with cd-rom)* (pp. 1867–1874). World Scientific.

Erhan, D., Bengio, Y., Courville, A., & Vincent, P. (2009). Visualizing higher-layer features of a deep network. *University of Montreal*, *1341*(3), 1.

Haag, A. V. (2012). Hercalibratie lobithw. *Deltares*.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the ieee international conference on computer vision* (pp. 1026–1034).

Henriksen, H. J., Troldborg, L., Nyegaard, P., Sonnenborg, T. O., Refsgaard, J. C., & Madsen, B. (2003). Methodology for construction, calibration and validation of a national hydrological model for denmark. *Journal of Hydrology*, *280*(1-4), 52–71.

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

Kratzert, F., Klotz, D., Brenner, C., Schulz, K., & Herrnegger, M. (2018). Rainfall-runoff modelling using long short-term memory (lstm) networks. *Hydrology and Earth System Sciences*, *22*(11), 9005.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097–1105).

Marçais, J., & De Dreuzy, J.-R. (2017). Prospective interest of deep learning for hydrological inference. *Groundwater*, *55*(5), 688–692.

Newman, A., Clark, M., Sampson, K., Wood, A., Hay, L., Bock, A., ... others (2015). Development of a large-sample watershed-scale hydrometeorological data set for the contiguous usa: data set characteristics and assessment of regional variability in hydrologic model performance. *Hydrology and Earth System Sciences*, *19*(1), 209–223.

Simonyan, K., Vedaldi, A., & Zisserman, A. (2013). Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.

Song, X., Zhang, G., Liu, F., Li, D., Zhao, Y., & Yang, J. (2016). Modeling spatio-temporal distribution of soil moisture by deep learning-based cellular automata model. *Journal of Arid Land*, *8*(5), 734–748.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, *15*(1), 1929–1958.

Tao, Y., Gao, X., Hsu, K., Sorooshian, S., & Ihler, A. (2016). A deep neural network modeling framework to reduce bias in satellite precipitation products. *Journal of Hydrometeorology*, *17*(3), 931–945.

Tieleman, T., & Hinton, G. (2012). Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, *4*(2), 26–31.

Wu, H., & Gu, X. (2015). Towards dropout training for convolutional neural networks. *Neural Networks*, *71*, 1–10.

Xingjian, S., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-K., & Woo, W.-c. (2015). Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems* (pp. 802–810).

Zhang, D., Lindholm, G., & Ratnaweera, H. (2018). Use long short-term memory to enhance internet of things for combined sewer overflow monitoring. *Journal of hydrology*, *556*, 409–418.