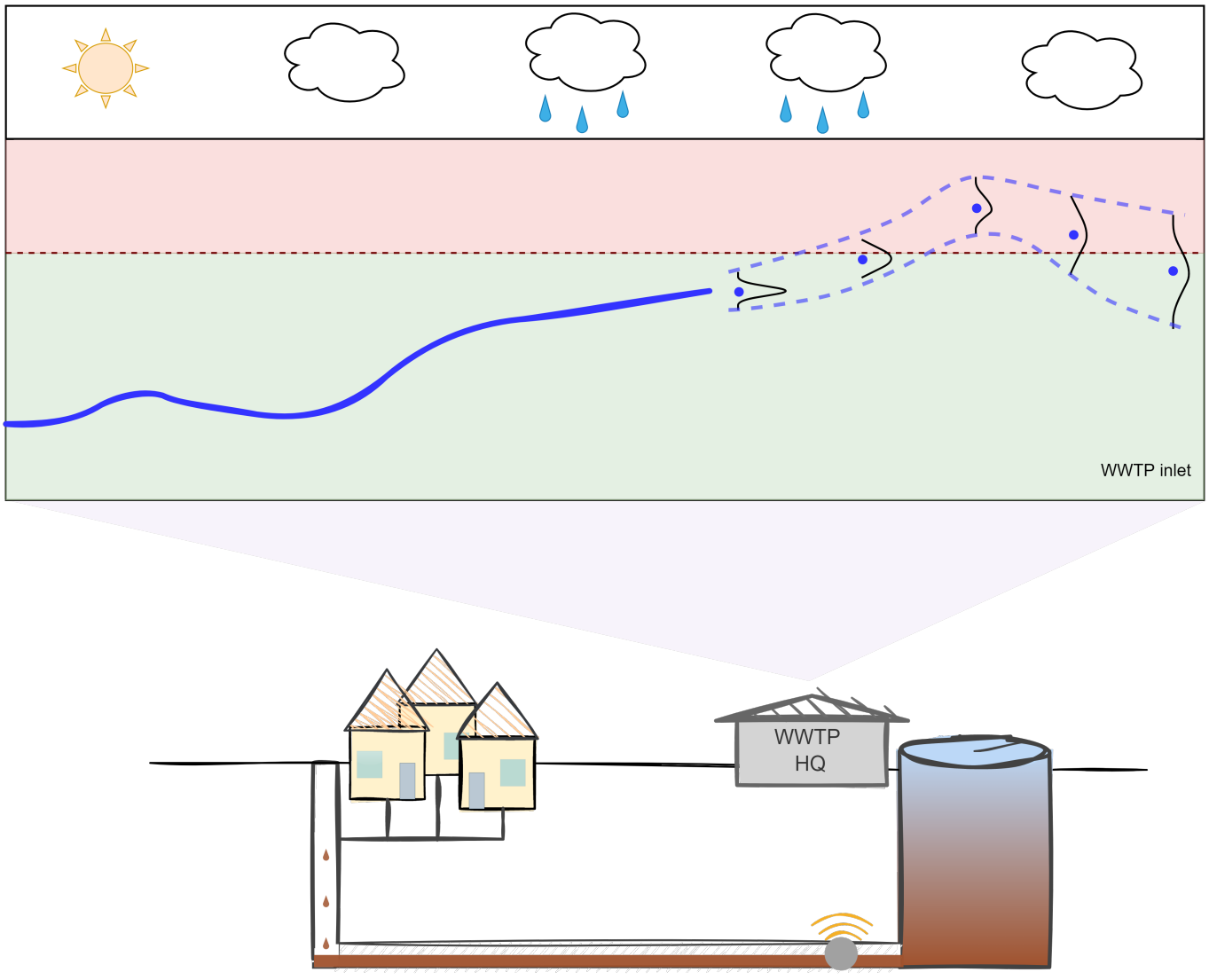
DTU Sustain
Department of Environmental and Resource Engineering

Probabilistic Machine Learning Models for Predicting Urban Drainage Flows

Master Thesis

Phillip Aarestrup, s163723

Supervised by:
Associate prof. Roland Löwe
Prof. Peter Steen Mikkelsen
Laura Frølich, Ph.d



Probabilistic Machine Learning Models for Predicting Urban Drainage Flows

Master Thesis
June, 2022

By
Phillip Aarestrup

Copyright: Reproduction of this publication in whole or in part must include the customary bibliographic citation, including author attribution, report title, etc.
Cover photo: Phillip Aarestrup, 2022
Published by: DTU, Department of Environmental and Resource Engineering, Bygningstorvet, Building 115, 2800 Kgs. Lyngby Denmark
https://www.sustain.dtu.dk/

Preface

This thesis has been prepared in a five months period, from January 24 to June 24 2022, at DTU Sustain, at the Technical University of Denmark, DTU. The thesis serves as a fulfilment for the degree Master of Science in Engineering, MSc Eng, and is the work equal to 30 ETCS points.

It is assumed that the reader has knowledge in the areas of statistics, machine learning and hydraulics.

In this thesis the following notations are used for equations:

- Small italic letters for scalars: x
- Small bold, italic letters for vectors: $\boldsymbol{x}$
- Large bold, italic letters for matrices: $\boldsymbol{X}$
- Large bold, italic, calligraphic letters for tensors: $\boldsymbol{\mathcal{X}}$

DTU Sustain, 24-06-2022

Phillip Aarestrup

Phillip Aarestrup

Abstract

This thesis develops proof-of-concept probabilistic forecasting models based on artificial neural networks. Specifically, the models are based on the long short-term memory cells to output a predictive distribution every 10 minutes with a forecast horizon of 3 hours. Different configurations of the models were tested. The general skill of the forecasts was tested, based on measured inflow to a wastewater treatment plant, using the smoothed persistence index (SPI), which measures the predictive skill relative to a benchmark. The continuous ranked probability score (CRPS) allows for comparing probabilistic and non-probabilistic forecasts. The skill to correctly forecast the exceedance of a critical flow threshold, which activates the wet weather operations (aeration tank settling, ATS), is measured by the critical success index (CSI). Missing data and anomalies are often a problem in operational settings. Hence the models were tested on their ability to fill in missing data under varying flow conditions and detect and replace anomalies.

This study shows that a proof-of-concept model without hyperparameter tuning has a similar or slightly worse performance than an optimized Box-Jenkins model found in the literature. The forecasted predictive probabilities yield similar results to using non-probabilistic models when comparing the general performance, with an SPI of 0.56-0.59. However, the probabilistic forecasts better describe the observed flow, shown by a lower CRPS of approximately 20-33% depending on the forecast horizon. Furthermore, both models can correctly predict the ATS-activation 80-50% of the time within 10-60 minutes. After the 120-minute forecast horizon, the performance decreases, corresponding to the catchment response time. This indicates that reliable forecasts can be made up until the response time of the catchment. When adding rainfall forecast to the model input, the predictive skill of the model increases for forecast horizons of 120 minutes or more, reducing the CRPS by additionally 25% and increasing the SPI by 0.12 and the CSI by a factor of 3 to 4. The models also proved stable when forecasting over extended periods without observations. They can thus be used for operational data imputation when data is missing, although with more uncertainty during high flow events. Regarding anomaly detection, a step is taken toward finding an automated system that detects and replaces anomalies. However, more research is needed to better represent the underlying distribution of the data in the model output to replace anomalies reliably.

Acknowledgements

I want to express my deep gratitude to my two supervisors Associate Professor Roland Löwe and Professor Peter Steen Mikkelsen, and external supervisor Laura Frølich. I have enjoyed our fruitful discussions during our weekly meetings; without them, the project would not have been possible. A special thanks should go to Roland Löwe for motivating me and giving me constructive feedback. A special thanks should also go to Peter Steen Mikkelsen for the motivation and asking the ”stupid questions” that got me re-assess the choices that I made throughout the project. I am also very thankful that Laura Frølich chose to continue as a supervisor, even though she did not have to; your inputs have been invaluable.

I would also like to thank DHI for allowing me to work on their project and BIOFOS, who showed great interest and came with constructive comments throughout our meeting.

I want to thank my friends and family, especially Clara, for tolerating me for five months while all my thoughts were centered around probabilities and distributions and for reading the thesis, correcting grammar, and improving the overall quality.

Finally, the past five months would not have been as fun without my amazing co-students, with whom I have had many interesting discussions - on serious as well as entertaining topics. So, thank you Freja, Magnus, Mathias, Oliver and Vinh.

Abbreviations

AE	absolute error.
ANN	artificial neural network.
ATS	aeration tank settling.
BPTT	back-propagation through time.
CCF	cross correlation function.
CDF	cumulative distribution function.
CNN	convolutional neural network.
CRPS	continuous ranked probability score.
CSI	critical success index.
CSO	combined sewer overflow.
DMI	Danish Meteorological Institute.
FFNN	feed-forward neural network.
FN	false negative.
FP	false positive.
HOFOR	Hovedstadens Forsyning.
LSTM	long short-term memory.
MCD	monte carlo dropout.
MSE	mean squared error.
NLL	negative log-likelihood.
NLP	natural language processing.
PDF	probability density function.
PI	persistence index.
ReLU	rectified linear unit.
RNN	recurrent neural network.
SDE	stochastic differential equation.
SGD	stochastic gradient descent.
SMOTE	synthetic minority oversampling technique.

SPI smoothed persistence index.
SSE sum of squared errors.
SVK Spildevandskomiteen.

TN true negative.
TOF time of forecast.
TP true positive.

UDS urban drainage system.

WWTP wastewater treatment plant.

Contents

List of Figures

List of Tables

1 Introduction

Urban drainage system (UDS) management becomes more challenged by the increasing urbanization (European Commission, 2016), changing rainfall patterns including more extreme precipitation events due to climate change (Arnbjerg-Nielsen, 2012), environmental regulations, and public interest (Rothenborg, 2022). Upgrading the integrated wastewater infrastructures is infeasible, both in terms of time and costs. Hence, the cities need to adapt to the growing demand by controlling the structures mentioned above and utilizing the increasing digitization (Eggimann et al., 2017; Lund et al., 2019).

In moving towards better management of the integrated urban wastewater infrastructures in the UDS, models with the ability to forecast flows and water levels are in demand. An example is the inflow to the wastewater treatment plant (WWTP), which is vital to forecast in advance to activate the wet-weather operation referred to as aeration tank settling (ATS) (Jørgensen et al., 1998) and thereby avoid bypass. Here, computation time is a crucial factor, considering that fast computation gives more time for the operator to take action. Furthermore, the forecast's uncertainty is also essential when making decisions. Thus, considered modeling approaches have a benefit if they can provide an estimate of the prediction uncertainty.

Two different regimes of models currently exist: physics-based models and data-driven models. Physics-based models have the advantage of being based on physical attributes of the catchment and system and simulate many points in the UDS at a chosen time resolution. The downside of the physical models is that they are slow and prone to errors in the physical attributes. Physics-based models can give uncertainty estimates. However, this requires running the models multiple times with different initial conditions or forecasts, e.g., ensembles. Data-driven models are a parameterization of the UDS trained on observed data and usually only predict a single point in a catchment. Data-driven models have the advantage of being extremely fast and accurate and can estimate the prediction uncertainty but are prone to errors in the data. Popular physics-based models are Mike+ (DHI, 2021) and SWMM (EPA, 2020), whereas data-driven models usually centers around Box-Jenkins models (Madsen, 2008), stochastic differential equations (SDEs) also called grey-box models (Breinholt et al., 2011; Carstensen et al., 1998; Thordarson et al., 2012), and now artificial neural networks (ANNs). For management and control operations, speed is a key factor when selecting a model; hence data-driven models have the advantage.

Previously grey-box models have been used to predict inflow to a WWTP based on rainfall input from rain gauges (Carstensen et al., 1998) and were developed further to better describe the uncertainty of the flow predictions (Breinholt et al., 2011). The predictions using grey-box models have also been expanded to include weather radar instead of rain gauges (Löwe et al., 2014) and predicting runoff volumes instead of flow (Löwe et al., 2016). However, the grey-box model framework is still limited since individual runoff processes need to be formulated in the model, and the modeling framework has proved

difficult to work with. In the past years, different types of ANNs have achieved state-of-the-art performance for various tasks. The first major breakthrough was in image recognition back in 2012 (Krizhevsky et al., 2012), and is still improving (Liu et al., 2021; Tang et al., 2021). Other fields have also seen a major improvement, e.g., text sentiment analysis (L. Zhang et al., 2018). Recently, ANNs have also entered the field of hydraulic (Bailey et al., 2016; Palmitessa et al., 2021; D. Zhang et al., 2018a,b) and hydrological modelling (Dong et al., 2020; D. Li et al., 2021; P. Li et al., 2022), which show promising results. Different types of ANNs exist, but commonly used are the feed-forward neural networks (FFNNs), which is the most simple type, and long short-term memorys (LSTMs), that are created to handle sequential data, such as time series (Hochreiter and Schmidhuber, 1997). Regardless of the choice of ANN, the framework offers a high level of flexibility compared to grey-box and Box-Jenkins models, particularly when working with more input data streams and different data types. The ANNs also offers an intuitive framework that is well documented and has a large and active community, which allows for fast model development and deployment.

Attempts to model the uncertainty of the predictions of the ANNs have also been tested. D. Li et al. (2021) use the LSTM network and a Bayesian Markov chain Monte Carlo (MCMC) to model the uncertainties in a three-step approach. Monte carlo dropout (MCD) in an ANN during inference has also been proposed as a way to mimic a Bayesian approach by creating a predictive distribution (Gal and Ghahramani, 2016). This technique has been used to estimate uncertainties in air quality sensors (Murad et al., 2021). However, using MCD requires multiple model predictions simultaneously to get an empirical predictive distribution, which can be used to obtain the uncertainty. Fang et al. (2020) modeled soil moisture across the United States by using a combination of MCD for estimating the uncertainty of the parameters, and an explicit output distribution of the predictions, for estimating the uncertainty of the inputs. None of the above-mentioned articles on ANNs work with multiple time steps in the forecast, and most do not disclose the architecture of the developed models.

In this thesis, I investigate if it is possible to obtain an ANN, which forecasts are predictive distributions of the inflow to the WWTP. I also test how the probabilistic models perform when compared to non-probabilistic. Furthermore, I test if the model can be used for filling missing time series or finding and replacing anomalies. This investigation leads to a novel and proof-of-concept ANN architecture that estimates the predictive distributions of the inflow to a WWTP with a time resolution of 10 minutes and a forecast horizon of 3 hours.

1.1 Research questions

The investigation can be split up into the following research questions:

1. How can we use an artificial neural network to forecast the inflow to the WWTP using readily available observations?
   (a) Which network architecture best suits the FFNN and LSTM for forecasting inflow depending on the output being probabilistic or non-probabilistic?
   (b) How accurate are the models forecasting the inflow to the WWTP at different forecast horizons?
2. Can the model be directly applied for filling in missing data or discovering anomalies?
   (a) To what extent can the model fill in missing data and use its predictions as input?
   (b) How can the probabilistic output of the model be used to find anomalies in the time series, and to what degree can the anomalies be replaced with model predictions?

1.2 Structure of the thesis

The thesis is structured into eight chapters. In this chapter, the aim of the thesis was introduced. In Chapter 2 the theory needed to understand the rest of the thesis is covered. Chapter 3 presents the case area of the study. Chapter 4 introduces the data used in the thesis. In Chapter 5 the methods used to obtain the results in Chapter 6, are described. The results are discussed, and a research outlook is given in Chapter 7. Finally, in Chapter 8 the main findings of the thesis are presented.

2 Background

Forecasting is essential when working with flow modeling in UDSs. Figure 2.1 simplifies a model forecasting the flow based on prior information of rainfall, measured flow, etc. The time the forecast is made is denoted t and referred to as the time of forecast (TOF). The forecast horizon, representing the number of time steps desired to predict ahead in time, is denoted h. Hence, a flow prediction, $\hat{y}$, at time t at horizon h is referred to as $\hat{y}_{t+h}$.

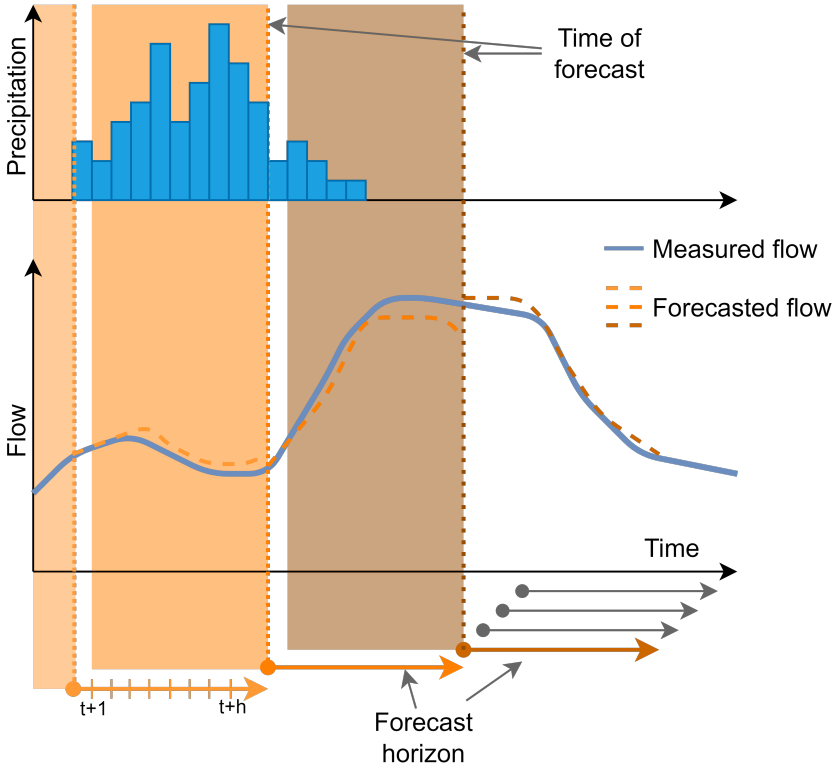


Figure 2.1: Conceptual drawing of flow forecasts. The blue line represents the measured flow, while the blue bars represent the measured precipitation. Dotted lines indicate time of forecast (TOF), dashed lines the forecasted flow, arrows the forecast horizon and shaded areas the observations available to the model at TOF, and grey arrows indicate a new forecast being created at each new time step.

Generally, the model forecast is given by Equation (2.1).

$$\hat{y}_{t+h|t} = f\left(\mathcal{X}_t, \hat{\theta}\right) \tag{2.1}$$

where $\hat{y}_{t+h|t}$ is the predicted flow at time step $t+h$, predicted at TOF t, and f is a function that maps the flow given observations $\mathcal{X}_t$ and calibrated parameters $\hat{\theta}$. For $h = 1$, the model seeks to predict the flow at the next time step. The observations, $\mathcal{X}_t$, given to the model represent previous data and are indicated by the shaded area in Figure 2.1. A model simultaneously outputting all predictions is referred to as a flow forecasting model throughout this thesis.

2.1 Probabilistic approaches

Within water management, uncertainty estimations are essential to make informed decisions. Due to computation time, physics-based models cannot be used in an online setting. Hence we rely on data-driven models. Following the increasing amounts of available data, data-driven models are now widely applied. Many data-driven methods currently exist, such as Box-Jenkins methods, Grey-Box models (SDEs) and now also ANNs.

Box-Jenkins methods rely on Auto-Regressive (AR) and Moving Average (MA) processes as well as additional terms like eXogenous inputs (ARMAX), differencing (Integrated, ARIMA), and Seasonality (SARMA) or a combination of those (SARIMAX). Common to these models is the need to identify an optimal set of parameters, maximizing the likelihood. Although the method is referred to as maximum likelihood, it is often turned into a minimization problem where the negative log-likelihood (NLL) is minimized. The models mentioned above can be calibrated using this approach under the assumptions that model residuals, ϵ, are identically and independent distributed (i.i.d), follow a white noise signal, and a normal distribution with constant variance σ_ϵ^2 and are based on the one-step-error prediction method (Madsen, 2008). The uncertainty is modeled from the estimated error variance and moving average coefficients and only depends on the forecast horizon. This also means that the Box-Jenkins models are only optimized to make one-step predictions and might diverge from the assumed normality for longer forecast horizons. The uncertainty estimates are calculated as shown in Equation (2.2).

$$q_h^{(\tau)} = u_{\tau/2}\sigma_\epsilon\sqrt{1+\psi_1^2+\ldots+\psi_{h-1}^2} \tag{2.2}$$

where $q_h^{(\tau)}$ is the τ-level quantile prediction interval at forecast horizon h, h is the forecast horizon, u_τ is the τ-level quantile of the standard normal distribution, σ_ϵ is the estimated error standard deviation, and ψ is the parameters of the moving average representation of the process. This equation gives both the upper and lower boundary of the prediction interval.

Grey-box models have also been proven to efficiently forecast flows in UDSs (Breinholt et al., 2011; Löwe et al., 2014). A set of equations describing the states and processes in the model must be defined to use grey-box models. The model is then optimized by maximum likelihood (see Breinholt et al., 2011). Multiple simulations of the same forecast horizon are calculated, resulting in an empirical probability density function (PDF) to obtain the model uncertainty (Thordarson et al., 2012). This is defined as seen by Equation (2.3).

$$q_{t+h|t}^{(\tau)} = F_{t+h|t}^{-1}(\tau) \tag{2.3}$$

where $q_{t+h|t}^{(\tau)}$ is the prediction interval for the τ-quantile forecast at the $t+h$ forecast horizon prediction at TOF t, $F_{t+h|t}^{-1}(\tau)$ is the inverse cumulative distribution function (CDF) of the multiple predictions $\hat{Y}_{t+h|t}$ at the $t+h$ forecast horizon at TOF t at the τ-quantile. The CDF have to be evaluated at both the upper and lower boundary.

When it comes to large data sets of multiple data sources, ANNs are advantageous, be-

ing highly flexible and allowing for efficient optimization of the numerous model parameters. Recently, several methods for probabilistic forecasting have been introduced, using monte carlo dropout (MCD) (Gal and Ghahramani, 2016) or directly estimating the forecast distribution by estimating the parameters describing the distribution. The latter is done by estimating the mean and variance for a normal distribution by a maximum likelihood (Barnes et al., 2021; Murad et al., 2021). Besides iteratively producing one-step forecasts, ANNs can also directly forecast multiple forecast horizons (Lim and Zohren, 2021; Makridakis et al., 2018; Wen et al., 2017).

All of the above methods, except MCD, are optimized by maximizing the likelihood of an observation being within the predicted distribution. The Box-Jenkins methods use one-step prediction errors to maximize the likelihood function (Madsen, 2008), grey-box models use Kalman Filter techniques and a Lamperti transform (Jazwinski, 2007; Thordarson et al., 2012). While ANNs directly evaluates the maximum likelihood from the normal distribution like the Box-Jenkins methods. However, for ANNs all horizons can be evaluated simultaneously and do not rely on the assumptions of the Box-Jenkins methods.

2.2 Artificial Neural Networks

An artificial neural network (ANN) represents many different types of network structures. Among these, the most common are the feed-forward neural network (FFNN), recurrent neural networks (RNNs) and convolutional neural networks (CNNs). Only the FFNNs and RNNs are applied in this thesis. The FFNN is composed of nodes and layers of nodes that, when combined, form the ANN. A node is a piece of the network that can apply a mathematical operation and is connected to other nodes via weights. When multiple nodes are present and get the same input, they form a layer. Each node takes an input and applies one or more weights and a bias to the input before applying an activation function. For most FFNNs a node is connected to all nodes in the previous layer and all nodes in the successive layer. An illustration of a simple FFNN can be found in Appendix A and an output from a layer can be calculated following Equation (2.4).

$$\boldsymbol{o} = a(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}) \tag{2.4}$$

where $\boldsymbol{o}$ is the output, $\boldsymbol{W}$ are the weights connecting the previous layer with the current layer, $\boldsymbol{x}$ is the input or the activation from the previous layer, $\boldsymbol{b}$ is the bias and a is the activation function (see Section 2.2.1).

In order to increase the complexity of a model and thereby allow it to learn more complex patterns, the model's width or depth can be increased by adding more neurons to a layer or adding more layers, respectively. However, increasing model complexity also increases the risk of overfitting.

2.2.1 Activation functions

Activation functions are used to obtain a non-linear relationship to the output of a neuron, allowing the mapping of non-linear relations between inputs and outputs of an ANN. Below are the three activation functions used in this thesis, Equations (2.5) to (2.7) shows the

sigmoid function, the hyperbolic tangent function, and the rectified linear unit (ReLU), respectively (Glorot et al., 2011; Nair and Hinton, 2010), which are further visualized in Appendix B. The output from the sigmoid function will lie between 0 and 1, the hyperbolic tangent between -1 and 1, and the ReLU be equal to 0 if the value is negative or equal to the value of the input.

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.5}$$

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.6}$$

$$\text{ReLU}(x) = \max(0, x) \tag{2.7}$$

where x is the input value to the activation function, and e is the exponential function.

2.3 Recurrent Neural Networks

FFNNs suffer from not being able to take into account any sequential structure of the data. RNNs solve this issue by saving features from each time step and thus be able to learn time dependencies. RNNs have been used in natural language processing (NLP) and machine translation among others, where they commonly outperform FFNNs, when working with sequential data (Alamia et al., 2020; Y. Bengio, 1991).

Optimization of RNNs is different than for FFNNs, since the error is propagated back through time. In back-propagation through time (BPTT) (Werbos, 1990), the error is not only propagated back through the hidden layers, but also through a time dimension. However, difficulties arise when training on longer sequences since RNNs often falls short when learning long-term dependencies, due to the vanishing (or exploding) gradient problem (Goodfellow et al., 2016). The problem happens when the same gradient is multiplied by itself many times during training, leading to it either vanishing or exploding. This problem could be avoided by keeping the model in a space where the gradients are stable, but Y. Bengio et al., 1993 showed that the optimization must enter a space where gradients eventually vanish or explode when using stochastic gradient descent (SGD). As the span in time steps increases, the challenge of training an RNN increases rapidly, and the probability of successfully training a network comes close to 0 for sequences with a length of 10 to 20 items (Y. Bengio et al., 1994).

2.3.1 Long Short-Term Memory

To avoid vanishing gradients, Hochreiter and Schmidhuber (1997) presented the LSTM cell which has shown superior performance when evaluated against the RNN. The structure of the LSTM is shown in Figure 2.2.

LSTM cells are designed to learn long-term dependencies using gating mechanisms that let them store information through multiple time steps. The overall structure of the LSTM cell is similar to that of the RNN considering that each recurrent neuron can be replaced by an LSTM-cell, of which the mathematical operations are quite different. Whereas the RNN only contains a hidden state, which is updated at each time step, the LSTM has a

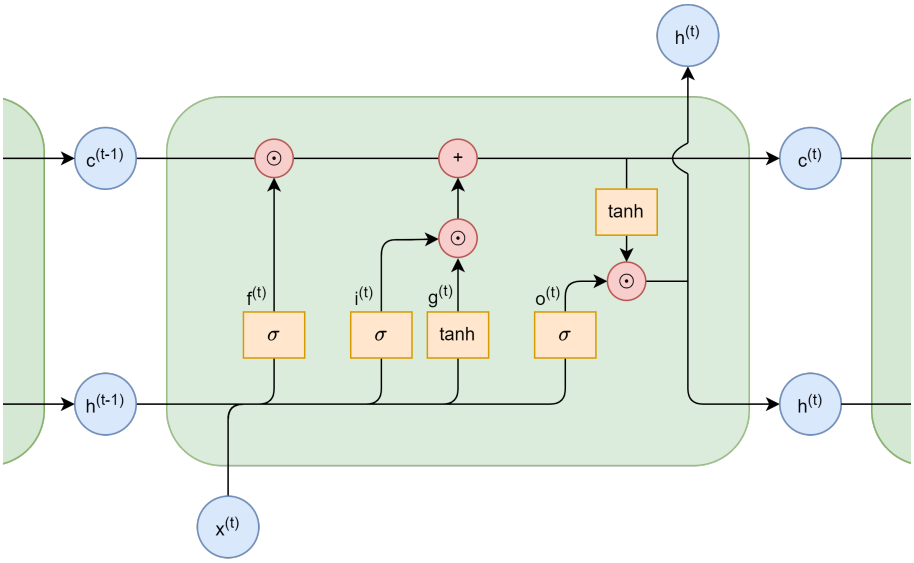


Figure 2.2: Overview of the LSTM cell. The hidden state $h^{(t-1)}$ and the sequence $x^{(t)}$ are inputted to the cell and through gating mechanisms the cell state $c^{(t)}$ is updated. Together, these can be combined to form the output of the cell $h^{(t)}$.

hidden state and a cell state, and both are updated. However, while the cell state is only updated through gating mechanisms, the hidden state is updated similarly to the RNN, allowing the LSTM to store information for longer periods in the cell state. The equations for updating both the hidden and the cell state are described in Equations (2.8) to (2.13).

$$\boldsymbol{f}^{(t)} = \sigma\left(\boldsymbol{W}_f \boldsymbol{x}^{(t)} + \boldsymbol{U}_f \boldsymbol{h}^{(t-1)} + \boldsymbol{b}_f\right) \tag{2.8}$$

$$\boldsymbol{i}^{(t)} = \sigma\left(\boldsymbol{W}_i \boldsymbol{x}^{(t)} + \boldsymbol{U}_i \boldsymbol{h}^{(t-1)} + \boldsymbol{b}_i\right) \tag{2.9}$$

$$\boldsymbol{g}^{(t)} = \tanh\left(\boldsymbol{W}_g \boldsymbol{x}^{(t)} + \boldsymbol{U}_g \boldsymbol{h}^{(t-1)} + \boldsymbol{b}_g\right) \tag{2.10}$$

$$\boldsymbol{c}^{(t)} = \boldsymbol{f}^{(t)} \odot \boldsymbol{c}^{(t-1)} + \boldsymbol{i}^{(t)} \odot \boldsymbol{g}^{(t)} \tag{2.11}$$

$$\boldsymbol{o}^{(t)} = \sigma\left(\boldsymbol{W}_o \boldsymbol{x}^{(t)} + \boldsymbol{U}_o \boldsymbol{h}^{(t-1)} + \boldsymbol{b}_o\right) \tag{2.12}$$

$$\boldsymbol{h}^{(t)} = \boldsymbol{o}^{(t)} \odot \tanh\left(\boldsymbol{c}^{(t)}\right) \tag{2.13}$$

where $\boldsymbol{W}_*$, $\boldsymbol{U}_*$, and $\boldsymbol{b}_*$ represent the input and recurrent weights, and biases, respectively with $_*$ denoting one of the four gates; the forget gate $\boldsymbol{f}^{(t)}$, the input gate $\boldsymbol{i}^{(t)}$, the update gate $\boldsymbol{g}^{(t)}$, and the output gate $\boldsymbol{o}^{(t)}$ where $^{(t)}$ is the time step. $\boldsymbol{c}^{(t)}$ and $\boldsymbol{h}^{(t)}$ represent the step, at which the cell state and the hidden state, respectively, are updated. As seen in Figure 2.2, $\boldsymbol{h}^{(t)}$ is both the hidden state and the output of each cell, at time t. $\boldsymbol{x}^{(t)}$ is the input at time step t. σ and tanh are the sigmoid and hyperbolic tangent activation functions, respectively. $\odot$ is the element-wise multiplication, also known as the hadamard product.

2.4 Training

Applying ANNs for forecasting implies training them on data by optimizing them from a loss function.

2.4.1 Loss functions

Before updates to the weights can be made, the errors between the predictions and observations are calculated by a loss function. For the deterministic models, the loss is calculated by the mean squared error (MSE) shown in Equation (2.14).

$$MSE = \frac{1}{TH} \sum_{t=1}^{T} \sum_{h=1}^{H} (y_{t+h} - \hat{y}_{t+h})^2 \tag{2.14}$$

where y_{t+h} and $\hat{y}_{t+h}$ is the measured and predicted observation, respectively, at time t and forecast horizon h.

The losses of the probabilistic model are calculated based on the likelihood. The losses are parameterized by the mean, μ, and the variance, σ^2, at each time step and forecast horizon as shown in Equation (2.15).

$$\mathcal{L}(\boldsymbol{x}_t) = \prod_{t=1}^{T} \prod_{h=1}^{H} \mathsf{Pr}(x_{t+h} | \mu_{t+h}, \sigma^2_{t+h}) \tag{2.15}$$

where $\mathcal{L}(\boldsymbol{x}_t)$ is the likelihood of the forecasted normal distributions for time steps $t+1$ to $t+h$ given a sequence $\boldsymbol{x}_t$ at time t. This equals the product of the probabilities when forecasting x_{t+h} at time $t+h$, parameterized by the predictive normal distribution $\mu_{t,h}$ and $\sigma^2_{t,h}$.

The product of a large series of probabilities, less than 1 will numerically become 0. Hence it cannot be used for optimization. This problem is solved by taking the logarithm to the likelihood, which turns the maximization problem into a minimization problem, thereby obtaining the NLL, ℓ. Taking the logarithm equates to taking the sums of the logarithm of the probabilities. The log probabilities are averaged across time and samples in a batch to avoid different scales of the gradients in batches. The NLL is shown in Equation (2.16).

$$\ell(\boldsymbol{x}_t) = -\frac{1}{TH} \log \mathcal{L}(\boldsymbol{x}_t) \tag{2.16}$$

2.4.2 Optimization

When the loss has been calculated from the batches, the parameters of the ANN are updated using a gradient descent algorithm. One such algorithm is the Adam algorithm, which is based on the adaptive estimates of the moments of the gradients. The algorithm have 4 tuneable parameters: η, β_1, β_2, and ϵ. η is the learning rate and defines the size of the step taken in the optimization, β_1 and β_2 are the exponential decay rates for the gradient's first and second moment estimates, and ϵ is a constant used for numerical stability.

During training, the learning rate is kept constant for the first 50 epochs and lowered using

exponential decay, as shown in Equation (2.17). If the learning rate is not lowered, it might be challenging to reach convergence since the step taken in the algorithm is too big to reach the basin of attraction, which is the minima of the loss landscape.

$$\eta(E) = \eta_0(1 - \lambda)^E \tag{2.17}$$

where $\eta(E)$ is the learning rate at epoch E, η_0 is the initial learning rate, and λ is the rate at which the learning rate decays.

Early stopping can be applied during training to avoid overfitting. Early stopping monitors the loss of the validation data set during training and stops the training before the model starts overfitting the training data. If the validation loss has not improved for several epochs, then the training is stopped.

3 Study area

This project focuses on the catchment of Damhusåen. The catchment is placed North-West of central Copenhagen and covers the area between Rødovre, Hvidovre, and Frederiksberg and continues up to Gladsaxe. The study area is shown in Figure 3.1 and covers a total area of approximately 37 km^2. The area has a population of 262,373 inhabitants as of 2015, and the population is expected to grow by 19% until 2025 (Thaileng, 2019), resulting in a potentially higher flow during both dry and wet weather conditions as a consequence of the possibly larger impervious area and the increased number of inhabitants. Furthermore, the UDS is made up of 85% combined sewer systems. This thesis aims to predict the inflow to the wastewater treatment plant (WWTP) placed in the southern part of the catchment.

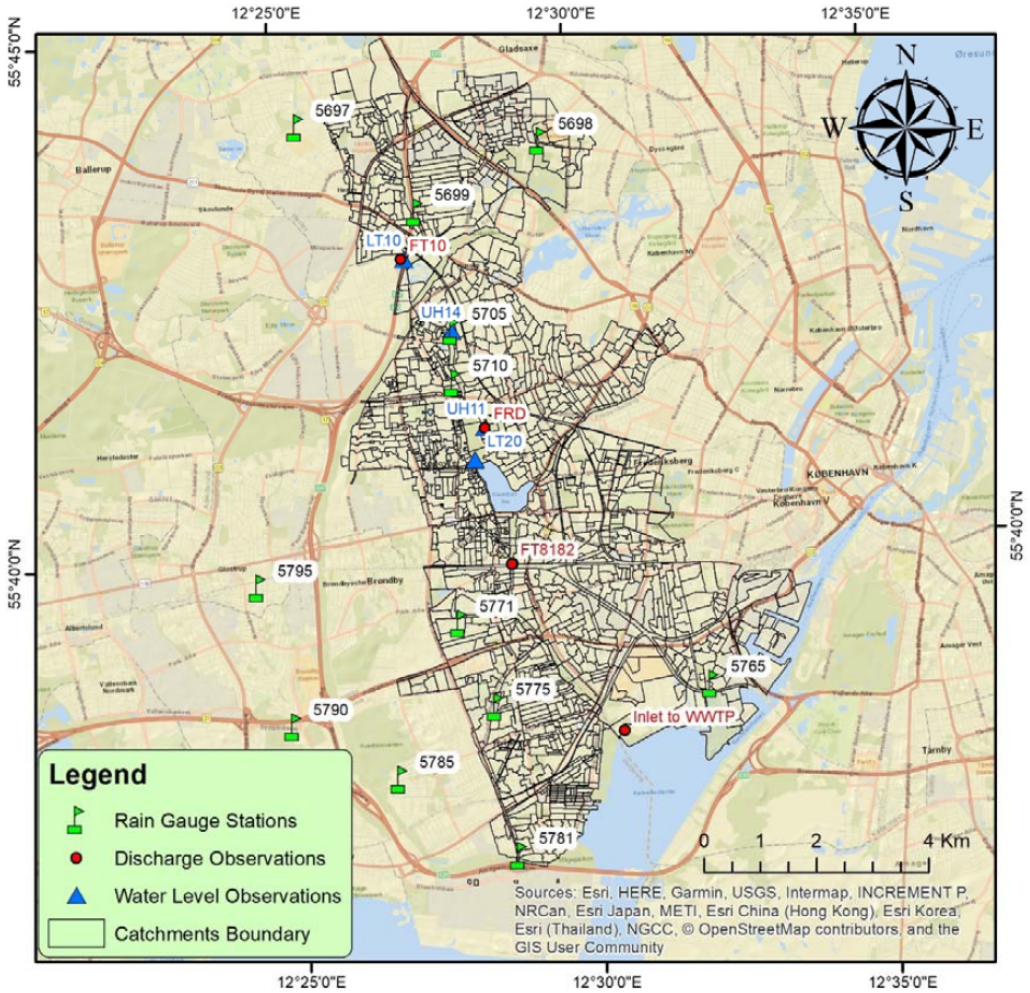


Figure 3.1: Map of the study area, showing locations for measurement stations and rain gauges (from Thaileng, 2019). Note: FT81 and FT82 are two individual sensors.

3.1 WWTP operations

The WWTP is built to treat wastewater under dry conditions, but when it rains, the plant's hydraulic capacity needs to increase. The hydraulic capacity is the flow rate at which the WWTP can still treat the wastewater. Hence, the overall operation of the WWTP can be split up into dry and wet conditions. The dry conditions are referred to as regular operation, whereas operation under wet conditions is called aeration tank settling (ATS) and is triggered when the flow reaches a predefined threshold. The ATS is what increases the hydraulic capacity. By increasing the hydraulic capacity of the WWTP, the risk of combined sewer overflow (CSO) or bypass is reduced by letting the sludge settle in the aeration tanks (Bundgaard et al., 1996; Nielsen et al., 1996; Sharma et al., 2013). Furthermore, the sludge's settling properties depend on multiple factors, such as age and temperature, and therefore the settling time differs between seasons (Sharma et al., 2013). The activation of ATS is critical; On one hand, an activation during dry conditions results in reduced treatment efficiency and, therefore, higher pollutant loads to the effluent. On the other hand, a delayed or missed activation of ATS leads to a higher risk of bypass, resulting in CSO. As mentioned in Sharma et al. (2013), the flow needs to be known in advance to turn on the ATS because it takes a while to adjust. In this case, the time needed to activate and get the ATS to run at optimal capacity is between 30 and 45 minutes in total (Jóhannesson et al., 2021). However, the exact time at which the beginning of the ATS event is predicted is not as important. As long as it can be predicted within 60 minutes before and 15 minutes after the beginning of the measured event. This is illustrated in Figure 3.2.

The activation of ATS can be turned into a binary classification setting for whether or not the flow is above the predefined threshold and if it is predicted within the set time window (see Section 5.3).

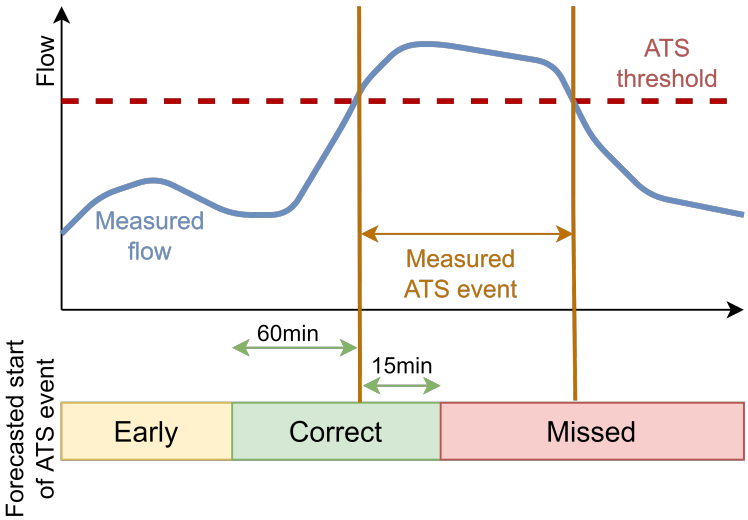


Figure 3.2: Illustration of when to activate the ATS based on the flow. To correctly forecast an ATS-event, the flow exceedance of the ATS threshold needs to be predicted within 60 minutes before or 15 minutes past the beginning of the measured ATS-event. (Adapted from Jóhannesson et al. (2021))

4 Data

The data section describes the data used for modeling and is a collection of water level and flow sensors and rain gauges. It also describes the procedures to clean and pre-process the data to make it ready for modeling.

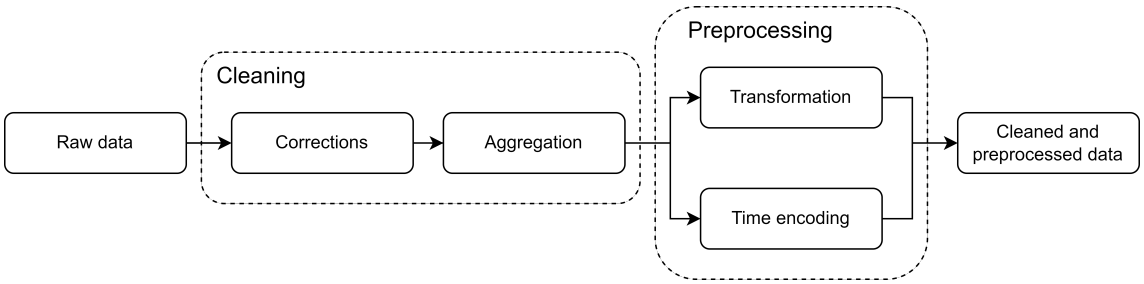


Figure 4.1: Flowchart explaining the process from raw data to readily formatted input data to the models. This includes cleaning and aggregating the data, and pre-processing it by transformations and time encodings.

The data section is split into three major steps as outlined in Figure 4.1.

Section 4.1 presents the data used for the modeling.

Section 4.2 describes how the data was cleaned and is divided into two parts:

4.2.1 Corrections made to the data in terms of flatline and spike removal.

4.2.2 Aggregation of the data to desired time resolution.

Section 4.3 defines how the data is pre-processed to alter existing features or obtain new, and consists of three parts:

4.3.1 Creating time encodings of the data.

4.3.2 Transforming the data.

4.3.3 A summary of the applied time encodings and transformations.

4.1 Data description

As seen on Figure 3.1, 12 rain gauges are placed inside and outside the catchment. Five flow sensors, as well as five water level sensors, are also available in the data set. The data goes back to 2012, but inflow data to the WWTP is only available from 2016. The inflow data to the WWTP is shown with the average precipitation of the 12 rain gauges in Figure 4.2. The rain gauges are a part of Spildevandskomiteen (SVK) and quality controlled by the Danish Meteorological Institute (DMI) (Jørgensen et al., 1998). The water level and flow sensors are operated by Hovedstadens Forsyning (HOFOR) and cleaned as described in Section 4.3. On Figure 4.2, the data is split into three sets referred to as the training, the validation, and the test set. These sets are used when training ANNs and further explained in Section 5.1.

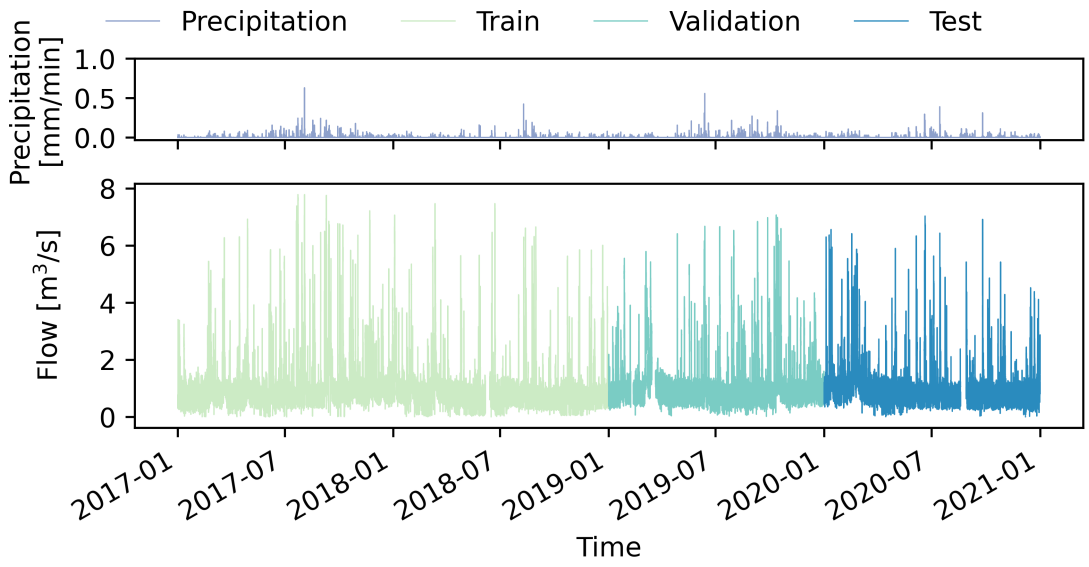


Figure 4.2: The data used for modeling, split up into training, validation and test periods. The precipitation is an average of the 12 rain gauges.

It can be seen from the data that the highest flows are contained within the training data set. Apart from that, the data flow patterns seem to be evenly distributed between the years, with slightly more events during the winter in 2020 and a long dry period during the spring in 2019.

4.2 Data cleaning

The data has previously been used for a master thesis (Jiang, 2021), where it has been cleaned and aggregated to the temporal resolution of 5 minutes.

4.2.1 Data corrections

Even though the data had been cleaned for apparent errors (see Jiang, 2021), periods where sensors are flatlining or spiking were found and portrayed in Figures 4.3 and 4.4.

Removing flatlines

To accommodate the issue of flatlines, differencing of the timeseries was performed. Equations (4.1) and (4.2) shows the terms used to identify the flatlining in form of total flatlining and linear increase or decrease, respectively.

$$D_t^1 = S_t - S_{t-1} \tag{4.1}$$

$$D_t^2 = |D_t^1 - D_{t-1}^1| \tag{4.2}$$

where D_t is the differencing of the data at time step t, S_t is the data point at time step t, and the superscripts ^{1,2} denotes the first and second differencing. The first differencing sets the value between flatlined values to 0 and the linearly increasing or decreasing values to a flatline. Hence, the second differencing will set these to zero. These new values can now be used to identify the two kinds of flatlines in the data. The second differencing is plotted against the measured flow in the top panel of Figure 4.3. A moving window sum is used

to identify the flatlines to find consecutive values of 0. Here, a window of $[-30\text{min}, 30\text{min}]$ was used setting values larger than 10^{-4} to 4 and else 0 (see Equation (4.3)), which is seen in the second panel of Figure 4.3. A moving window minimum is used to remove values where the minimum value of the window is zero to avoid having too many small periods of missing data (see Equations (4.4) and (4.5)). A window of $[-210\text{min}, 210\text{min}]$ was chosen since this is the required sequence length to run the network. The result is shown in the third panel and the cleaned time series in the fourth panel of Figure 4.3.

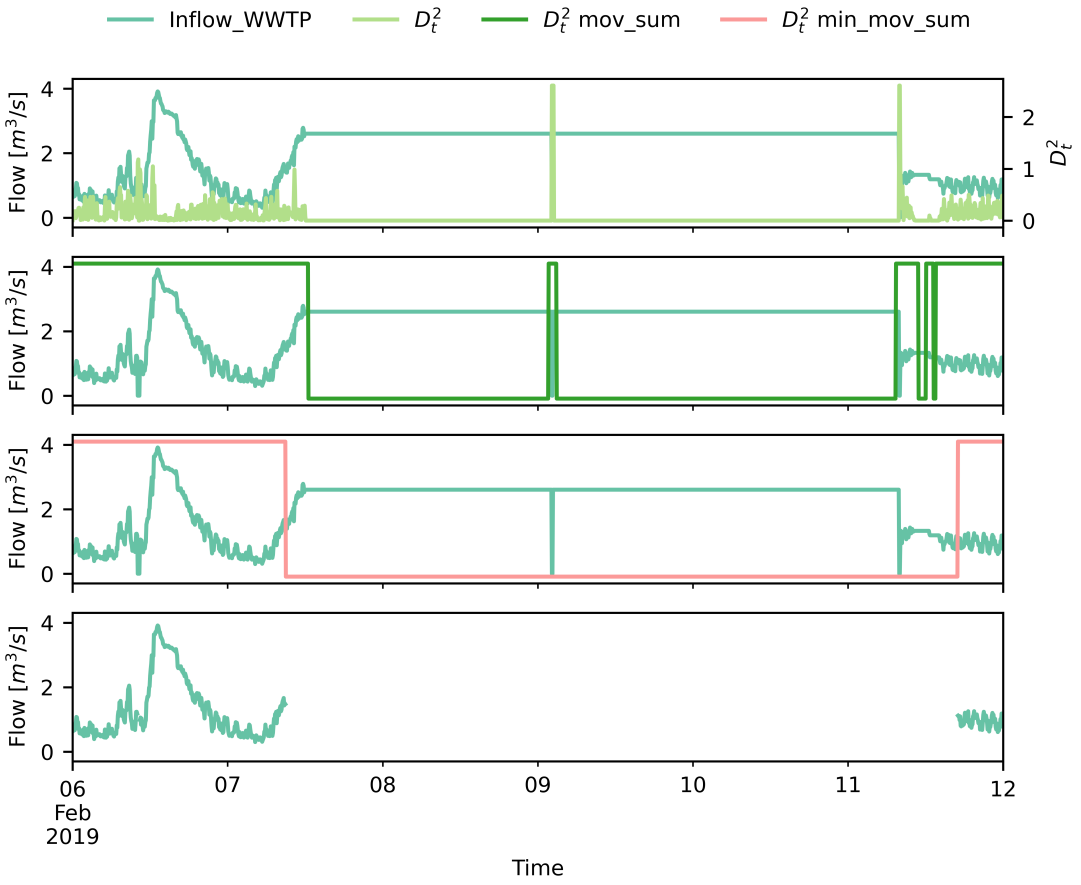


Figure 4.3: An example of a flatline in the time series. The first panel shows the flow plotted alongside the second differencing. The second panel shows the flow plotted against the moving sum, which sets values above 10^{-4} to 4 and else 0. The third panel shows the flow plotted against the moving minimum, which removes the values where the minimum value inside the window is 0. The fourth panel shows the cleaned time series.

$$D_t^2 ms = \begin{cases} 0 & \text{if } \sum_i D_{t+i}^2 \leq 10^{-4} \\ 4 & \text{otherwise} \end{cases} \qquad (4.3)$$
$$\text{where} \quad i \in [-30[min], 30[min]]$$

$$D_t^2 mms = \min[D_{t-210}^2 ms, D_{t-200}^2 ms, \ldots, D_{t+200}^2 ms D_{t+210}^2 ms] \tag{4.4}$$

$$x_t = \begin{cases} \text{keep} & \text{if } D_t^2 mms \geq 0 \\ \text{remove} & \text{otherwise} \end{cases} \tag{4.5}$$

Removing spikes

In spike detection, the differencing from Equation (4.1) is applied again. A threshold is then chosen to mark values above the threshold as spikes. Here a threshold of $2.3\frac{m^3}{s}$ has been used, corresponding to the 99.99% percentile of all data differences to keep as much data as possible. As seen in Figure 4.4, a spike can consist of more than one value, but only one is marked. Hence the data points within 10 minutes of a marked spike are removed. Values are linearly interpolated if available within 20 minutes of the marked spike to avoid losing too much information. This interpolation will create small patches of linearly increasing or decreasing values but will be easier to fit for the model and closer to the actual flow.

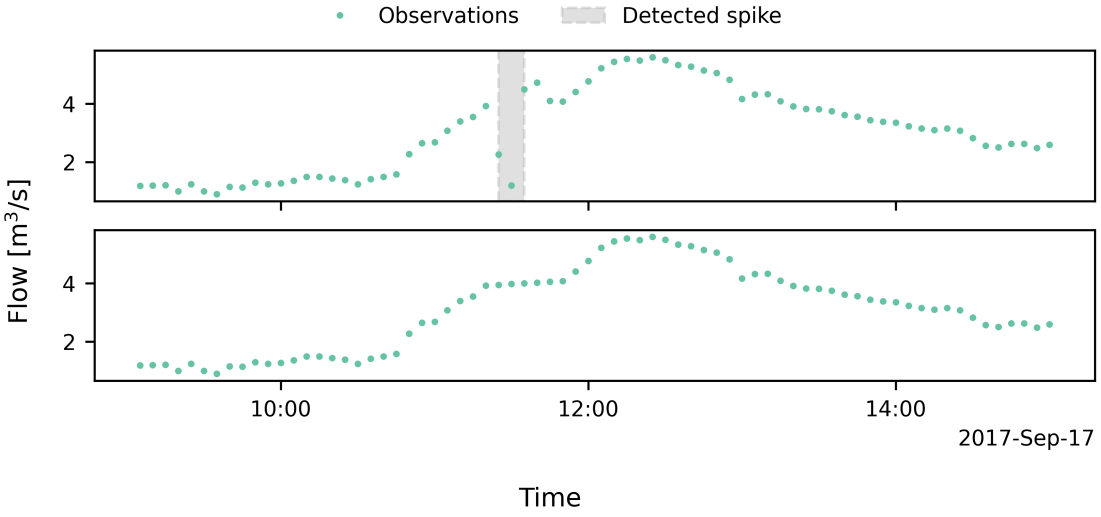


Figure 4.4: The first panel shows an example of a detected spike. The second panel shows the linear interpolation of the time series after spike is removal.

Since the context of the neighboring observations determines a spike, a median filter could also have been used. However, in this case, the spike is also based on a threshold, and selecting this threshold would require human intervention. Differencing seems to work in this case and is easier to work with; hence this is chosen.

4.2.2 Data Aggregation

The data is received with different time resolutions. Originally rain intensity, flow, and water level measurements all come in a 1-minute resolution. However, the received resolution was 5 minutes for rain intensities. The data were aggregated to a 10-minute resolution using the average over the 10 minutes. Data aggregation is needed because inputs to

the models have to be in the same temporal resolution. 10 minutes were chosen since this is the typical resolution for forecast data.

4.3 Pre-processing

Before a network was trained, data transformations and time encodings were performed on the cleaned data.

4.3.1 Time encoding

Before the model can utilize the timestamps, these must be encoded into features. In this thesis, the time is mapped onto a unit circle to represent daily and weekly patterns. The encoding consists of a sine and cosine transformation and are calculated using the generic formulas in Equations (4.6) and (4.7). The unit circle mapping is applied instead of a raw hour of day or week numbering since it ensures that the series is continuous. Using raw hour numbers will create discontinuity when going from 24 to 0.

$$X_{t,\text{sin}} = \text{sin}\left(2\pi \frac{t_{\text{unix}}[\text{s}]}{res[\text{s}]}\right) \tag{4.6}$$

$$X_{t,\text{cos}} = \text{cos}\left(2\pi \frac{t_{\text{unix}}[\text{s}]}{res[\text{s}]}\right) \tag{4.7}$$

where $X_{t,\text{sin}}$ is the encoded time feature for X in timestep t using the sine, t_{unix} is the unix time of t in seconds and res is the wanted resolution in seconds. For example, if encoding daily timesteps, res is chosen to 86,400 seconds since this is the number of seconds in a day. When the sine and cosine encodings are combined, they describe the time feature at the chosen resolution.

In this thesis, daily and weekly features are encoded to account for diurnal flows and weekly changes, e.g., higher flows during rush hours on the weekdays and potential changes in patterns between weekdays and weekends.

4.3.2 Transformations

Transformations are often used in time series analysis to scale the time series such that extreme values come closer to normal distributed values, as extreme values are impacted more by the transformations. For example, taking the logarithm to a log-normal distribution gives a more ”natural” distribution. In this work, the data can be transformed with the logarithm of base 10 as seen in Equation (4.8).

$$X_{\text{log}_{10}} = \text{log}_{10}(X) \tag{4.8}$$

where $X_{\text{log}_{10}}$ is the log_{10}-transformed feature X. This transformation is only applied to the flow as shown in Table 4.1.

Min-max scaling is also applied to avoid input values that are different by multiple magnitudes. This scaling is needed as features with higher values also produce bigger gradients during training, resulting in unstable training or training that does not converge. This scaling method transforms each feature to be within a defined range, which for this thesis is $[0, 1]$, and is calculated by Equation (4.9).

$$X_{[min_r, max_r]} = \left(\frac{X - \min(X)}{\max(X) - \min(X)} \right) (max_r - min_r) + min_r \tag{4.9}$$

where $X_{[min_r, max_r]}$ is the scaled version of a feature X, and min_r and max_r are the lower and upper boundaries of the range, i.e. 0 and 1, respectively. This transformation is applied to all time series as indicated in Table 4.1. The scaling based on the training set is applied to both the validation and test set to avoid information leakage between the sets using the $\min(X_{\text{train}})$ and $\max(X_{\text{train}})$.

The $\log_{10}$ transformation is only used if mentioned explicitly and is applied as a first step after which the min-max scaling is applied whether or not the data has been $\log_{10}$ transformed.

4.3.3 Final set of input variables

The final set of input variables and the applied transformations are shown in Table 4.1. The inputs are the inflow to the WWTP, the rain intensity averaged over the 12 rain gauges at each time step (Rain avg.), and the time of day (TOD) and time of week (TOW) encodings.

Table 4.1: The transformations applied to each time series in the data. TOD is the encoded time of day and TOW is the encoded time of week for either the sine of cosine transformation.

	$\log_{10}$	min-max
WWTP_inflow	(X)	X
Rain avg.		X
TOD_sin		X
TOD_cos		X
TOW_sin		X
TOW_cos		X

5 Methods

The methods section covers the modeling setup and how the best model combination is achieved. It also describes how a trained model can be applied to fill in missing data or find anomalies. Lastly, the metrics used to evaluate the models are explained.

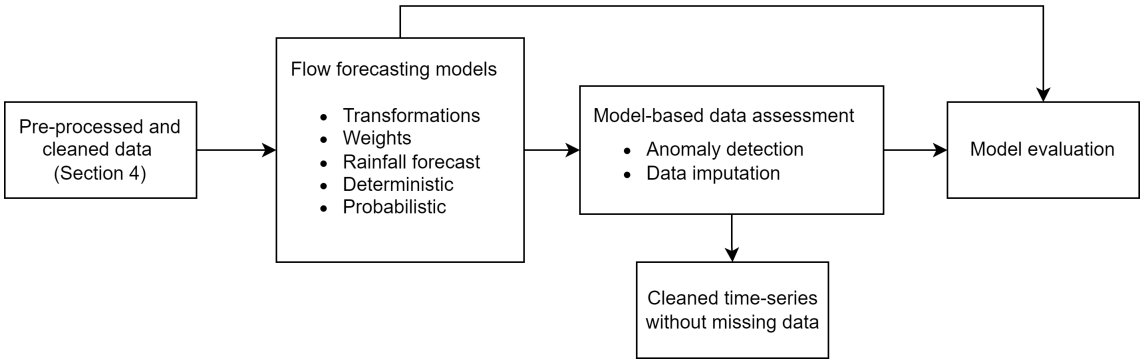


Figure 5.1: Flowchart describing the workflow from having pre-processed and clean data to evaluating the models or obtaining an automated cleaned time series based on model predictions.

The thesis proceeded in three major steps, as outlined in Figure 5.1

Section 5.1 describes the forecasting model and gives an overview of the combinations that are tested. The section is split up into model structures and training techniques ordered by section:

- 5.1.1 Model architecture based on point-prediction or probabilistic output
- 5.1.2 Training with sample weights to account for a skewed data distribution
- 5.1.3 Using rainfall forecast in the input to account for possible future increases in the flow due to rainfall
- 5.1.4 Summary of the tested model combinations

Section 5.2 introduces two additional use-cases of the probabilistic model where it can be used in an automated framework to do data imputation and anomaly detection which is split into three sections:

- 5.2.1 Using one-step predictions from forecasts
- 5.2.2 Replacing missing observations using the predictions of the trained neural network
- 5.2.3 Marking or replacing anomalies in data based on previous observations

Section 5.3 explains the evaluation metrics used to assess the models in this thesis.

5.1 Flow forecasting models

The illustration in Figure 5.2 shows the conceptual architecture of the models. At each time step, sequences of input variables are transformed into a sequence of forecast values of 18 time steps (180 minutes) into the future. In the figure, input sequences from time step $t - P$, where P is the number of time steps to go back, are given to the model, and the model predicts the outputs from $t+1$ to $t+H$ simultaneously, where H is the number of time steps in the forecast horizon. P and H are 24 and 18, respectively. The input to the models can vary, and selected inputs and outputs are shown in Figure 5.2, with the inputs being the flow, average rain intensity of the rain gauges, the encoded time of day (TOD), and time of week (TOW) (see Section 4.3.1) for the last 4 hours and potentially forecasted rain sequences, if included. Depending on the possible inputs used, it results in between 24 and 6 inputs per time step, whether forecasts are included or not. The figure also illustrates the LSTM model approach, where the states are updated at each time step. The FFNN model approach is shown in Appendix C, and takes the whole input and maps it to the output.

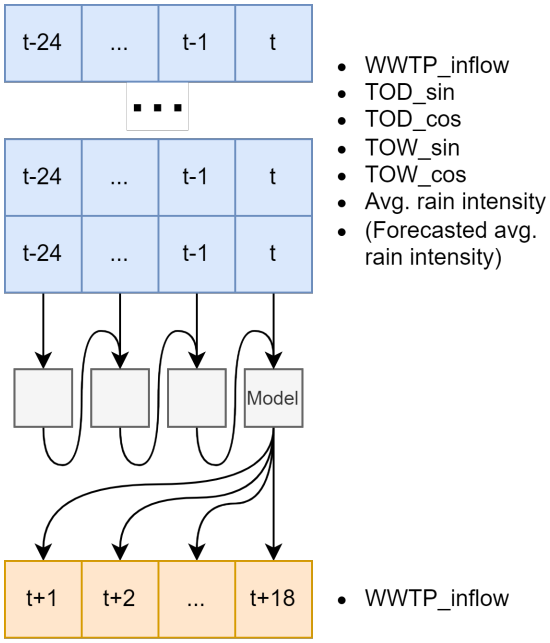


Figure 5.2: Overview of a LSTM model. The model receives inputs from the previous $[t-24, \ldots, t-1, t]$ observations and predicts the full output sequence $[t+1, t+2, \ldots, t+18]$ simultaneously. The time series used for the inputs and the predicted output is also shown.

Different initial strategies were tested in multiple combinations to establish a proof-of-concept model. The considered strategies were a model that:

- have a probabilistic architecture and output (P)
- have a log-transformation of the flow data (T)
- are trained with sample-weights (W)
- have a rainfall forecast in the input (RF)

- use a combination of the above

The explanation of the combinations can be found in Sections 5.1.1, 4.3.2, 5.1.2 and 5.1.3, respectively. For each of these combinations, 5 models were trained since a level variability exist in each training, resulting in 50 models. The combinations can be found in Section 5.1.4 in Table 5.2.

To train the ANN, the data is split up into a training, validation and testing set, shown in Figure 4.2. Here, data from 2017-2019 is used for training, data from 2019-2020 is used for validation, and data from 2020-2021 is used for testing.

The training set is mainly used for fitting the model parameters during the model training. The validation set is used to monitor the loss during the training epochs and be able to stop the model if it starts to overfit the data, that is, if the loss on the validation set starts to increase. Furthermore, the validation set is used to select the best architecture and tune the hyper parameters of the model. The test set is held out during training and model selection, and is used only to test the generalization performance of a final model (Y. Bengio, 2012).

In this thesis, during training, the Adam algorithm is chosen because it is one of the most commonly used stochastic gradient descent (SGD) algorithms and shows overall good performance (Kingma and Ba, 2014). The standard values as defined in *Tensorflow 2* are used. Furthermore, exponential decay of the learning rate was used with a decay rate of 0.005, meaning that the learning rate is lowered by 0.5% at each epoch. Besides exponential decay of the learning rate, early stopping is also applied. In this project, the model is stopped if the validation loss does not improve for 20 epochs.

Dropout was also applied as a regularizer to avoid overfitting the ANN (Srivastava et al., 2014). Dropout works by deactivating a neuron or output with a given probability during model training, and forces the ANN to not rely on a single path in the model. The dropout rate is a hyper-parameter that can be tuned. Here, 0.25 was used.

5.1.1 Probabilistic architecture and output

The architecture of the models followed one of two structures depending on the model being probabilistic or non-probabilistic. They are portrayed in Figure 5.3. In the figure, the LSTM versions of the networks are shown, while the FFNN versions can be found in Appendix C.

The probabilistic model receives an input $\mathcal{X}$ and two LSTM pipelines of which one estimates $\hat{\mu}$ and the other estimates $\hat{\sigma}$ of the normally distributed output, were trained in parallel based on the same input. The output from the LSTM layer is determined by the latent dimension of the LSTM-cell, here 50. This output is followed by a dropout layer, which is only used during training and connected to a linear layer outputting the length of the predicted sequence, here 18 (corresponding to 3 hours with 10-minute time steps). The estimated means and standard deviations are combined to form a normal distribution at each forecast horizon. The non-probabilistic model only has a single pipeline, similar to estimating the means in the probabilistic architecture. However, instead of the output

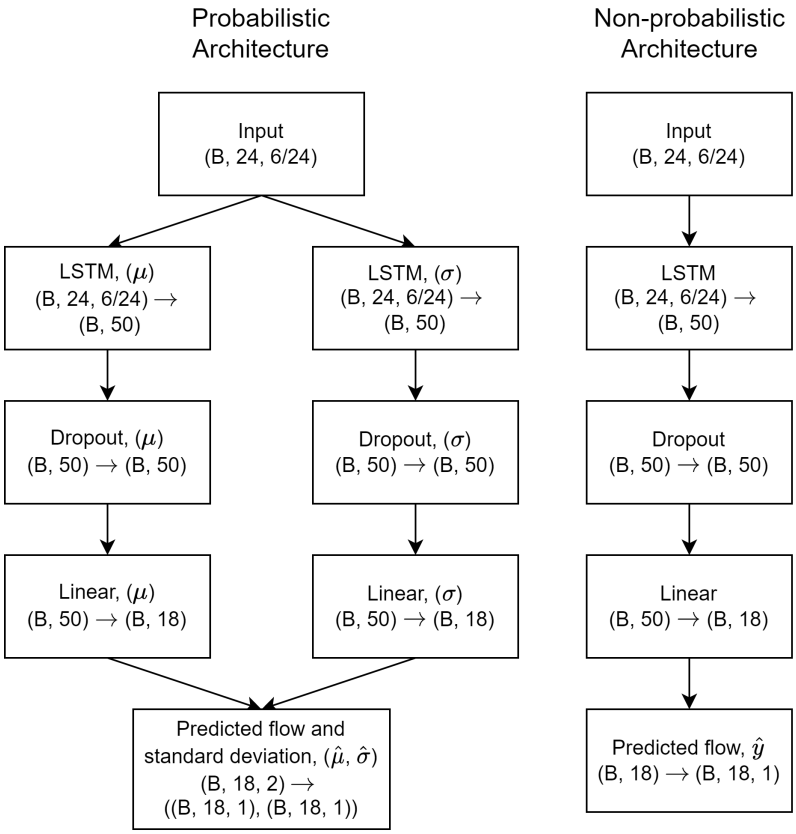


Figure 5.3: Flow forecasting model architecture for the probabilistic and non-probabilistic models. The probabilistic model uses two parallel LSTMs that predict $\hat{\mu}$ and $\hat{\sigma}$, respectively, for the normal distribution, whereas the non-probabilistic model only has a single pipeline predicting the output $\hat{y}$. For simplicity reshaping layers are omitted from the figure, but can be found by the shapes going in and out of the layers in the format (input shape) $\rightarrow$ (output shape) shown below the layer name. For example, (B, 24, 6/24) corresponds to a tensor with dimensions B (batch size), 24 (length of the input sequence), and 6 or 24 (number of inputs with or without rainfall forecast).

from the linear layer taking the form of a normal distribution, it outputs a point prediction for each of the forecasted time steps.

The two network types were trained differently, with the probabilistic one being trained by minimizing the NLL and the non-probabilistic one by minimizing the MSE. An explanation of these loss functions are found in Section 2.4.1.

5.1.2 Sample weights

Flow and water level data in urban drainage settings are highly skewed. Hence the ANN might have difficulties predicting extreme flows because fewer observations are available. Assuming that the data can be split into different classes of observations, i.e., dry-weather/wet-weather, there are multiple ways of alleviating this problem. The most feasible way to account for the data distribution is by weighting the individual samples in the loss function during training. The weights are calculated by binning the training data into five equally sized bins. The number of samples in each bin is then divided by the total number of samples in the training data. Then the fraction is subtracted from 1,

such that the smaller classes are assigned a larger weight. The procedure is shown in Equation (5.1), assuming binned data.

$$w_{Q,b} = 1 - \frac{N_{Q,b}}{N_Q} \tag{5.1}$$

where $w_{Q,b}$ is the weight assigned to flow samples in bin b, $N_{Q,b}$ is the number of flow samples in bin b, and N_Q is the total number of flow samples in the data. The weighting was done for both the min-max scaled data and the data which was both $\log_{10}$-transformed and min-max scaled. The resulting weights can be found in Table 5.1.

Table 5.1: Sample weights for the 5 bin-classes for normal data (Table 5.1a) and $\log_{10}$-transformed data (Table 5.1b). Note, that data is min-max scaled.

Bin interval	w_b	Sample counts
[0-0.2]	0.135	90 888
[0.2-0.4]	0.931	7 223
[0.4-0.6]	0.980	2 063
[0.6-0.8]	0.992	794
[0.8-1]	0.998	262

(a) Sample weights for the normal data

Bin interval	w_b	Sample counts
[0-0.2]	0.355	65 246
[0.2-0.4]	0.651	35 316
[0.4-0.6]	0.994	639
[0.6-0.8]	0.999	20
[0.8-1]	0.999	9

(b) Sample weights for $\log_{10}$-transformed data

The tables show that the weights do not sum to one as more than two bins are present. If this were desired, a re-scaling of the weights would be needed. However, this should not affect the optimization since the scale of the bins is the same. From Table 5.1a, it also becomes clear that the data is very skewed since 90% of the data is part of the first bin corresponding to dry weather flow, whereas less than 1% of the data is in the last bin, corresponding to heavy precipitation. When $\log_{10}$-transforming the data, more data points move into the second bin as seen in Table 5.1b, making the data more normal distributed than before the transformation. But, the last three bins now have fewer observations than before the transformation.

For a more in-depth discussion of the other possibilities to accommodate skewed data, see Section 7.2.2.

5.1.3 Rainfall Forecast

It was expected that the model needed precipitation as input to forecast peak flows correctly. Therefore, including forecasted rain intensity in the model should increase its predictive skill. Here, a perfect forecast was used, being the average rain intensity of the rain gauges. Using the perfect forecast will lead to less uncertainty than an actual forecast. However, it will show if the performance can be increased. Using an actual forecast would be expected to lie in between the skill of using a perfect forecast and not using a forecast (Löwe et al., 2014).

The perfect forecast was made by lagging the average rain intensity at time t by h, where h is the horizon being forecasted, giving rise to a new sequence with time reference $t+h$.

5.1.4 Model combinations

To find a model that can be established as proof-of-concept model, different combinations of the methods described above were tested. Below, in Table 5.2, the tested combinations are summarized.

Table 5.2: Combinations of the flow forecasting models. P: probabilistic, T: $\log_1 0$-transformed, W: weights applied, RF: rainfall forecast included.

Name	Probabilistic	$\log 10$-transformation	Sample weights	Rainfall forecast
FFNN				
FFNN-P	X			
FFNN-PT	X	X		
FFNN-PW	X		X	
FFNN-PTW	X	X	X	
FFNN-P-RF	X			X
LSTM				
LSTM-P	X			
LSTM-PT	X	X		
LSTM-PW	X		X	
LSTM-PTW	X	X	X	
LSTM-P-RF	X			X

5.2 Model-based data assessment

A trained model can assess the data from a model perspective, for example filling in missing data. Missing data is an enormous problem for all models, especially in data-driven models, since it would lead to wrong predictions or no forecasts because the input drives the output. In the case of the models used in this thesis, no forecast is made if inputs are missing. Hence, it is likely that the model skill could be improved by imputing missing values, and for this, a trained model could be applied. Furthermore, the model can also detect anomalies in the time series. The anomalies could stem from malfunctioning sensors, a clogging somewhere in the system, or emptying of filled storage.

This section is structured in 3 subsections. First, a discussion of the forecast horizon used for imputing missing values and evaluating anomalies is found in Section 5.2.1. Then, the framework for imputing missing data with the model is presented in Section 5.2.2. Finally, the anomaly detection strategy is introduced in Section 6.3.2.

5.2.1 One-step predictions

When the model is forecasting, it produces the whole 3-hour forecast horizon. However, when imputing data or finding anomalies, only the most reliable prediction horizon is needed, which is the prediction closest to the measured observations, $\hat{y}_{t+1}$, as shown in Equation (5.2). Thus, although the whole forecast is produced, only the one-step predictions are used to impute data or determine if observations are anomalies.

$$\hat{y}_{t+1}, [\hat{y}_{t+2}, ..., \hat{y}_{t+h}] = f\left(\mathcal{X}_t, \hat{\theta}\right) \tag{5.2}$$

where $\hat{y}_{t+h}$ is the flow predicted at time of forecast (TOF) t at a given forecast horizon h, [ ] indicate that these predictions are not used, f is the ANN mapping the input $\mathcal{X}_t$ to the output with the calibrated parameters $\hat{\theta}$.

5.2.2 Data imputation

The model was created not to make any predictions if data is missing in the input since this would lead to faulty predictions due to missing activations in the ANN, and the model was not trained to handle potential missing observations. Hence, if data is missing in the input, critical predictions of ATS events could be predicted too late or entirely missed. Therefore, a need for imputation of missing data is needed. The imputation of missing data could also solve the general issue of missing data in sensor time series.

Imputing missing data

The model can be used to impute the missing values for a given horizon. A forecast made at TOF t, forecasting the sequence $[\hat{y}_{t+1}, \hat{y}_{t+2}, \ldots, \hat{y}_{t+h}]$, can in theory impute missing values along the whole forecast horizon, however, the increase in forecast horizon also means an increase in uncertainty. Therefore, only the prediction in time steps $t+1$ was used to fill in a missing observation as described in Section 5.2.1. When the missing observation is replaced with the predicted value, the TOF at time $t+1$ can be made using the imputed value, meaning that the predicted value becomes part of the input data. The imputation can thus be done iteratively. The method is explained in the flowchart in Figure 5.4.

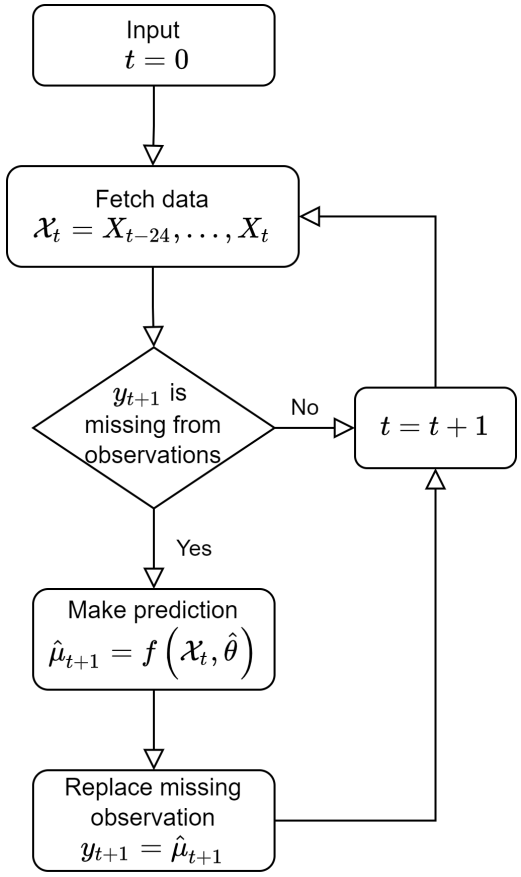


Figure 5.4: Flowchart describing the process of imputing a missing observation. X_t is the data at time t, y_{t+1} is the observed value, $\hat{\mu}_{t+1}$ is the mean of the predicted distribution and $f\left(\mathcal{X}_t, \hat{\theta}\right)$ is the model mapping the input data to the predicted distribution, where only the mean is used.

Since the probabilistic model contains a normal distribution of the prediction, $\mathcal{N}(\hat{\mu}, \hat{\sigma})$, based on the predicted mean, $\hat{\mu}$, and standard deviation, $\hat{\sigma}$, the missing observation could be imputed in three ways. One way of replacing the missing observation is to use the mean of the predicted distribution since this is the value with the highest probability. The second way is to use a quantile of the distribution. However, this only makes sense if the model generally under- or overestimates or in specific flow regimes. The third way is to sample from the distribution, which would give the imputation stochasticity. In this thesis, for simplicity, the mean was used for imputation. An illustration of how the imputation was made

is shown in Figure 5.5, where a missing value y_{t+1} is replaced with the predicted mean, $\hat{\mu}_{t+1}$ of the distribution.

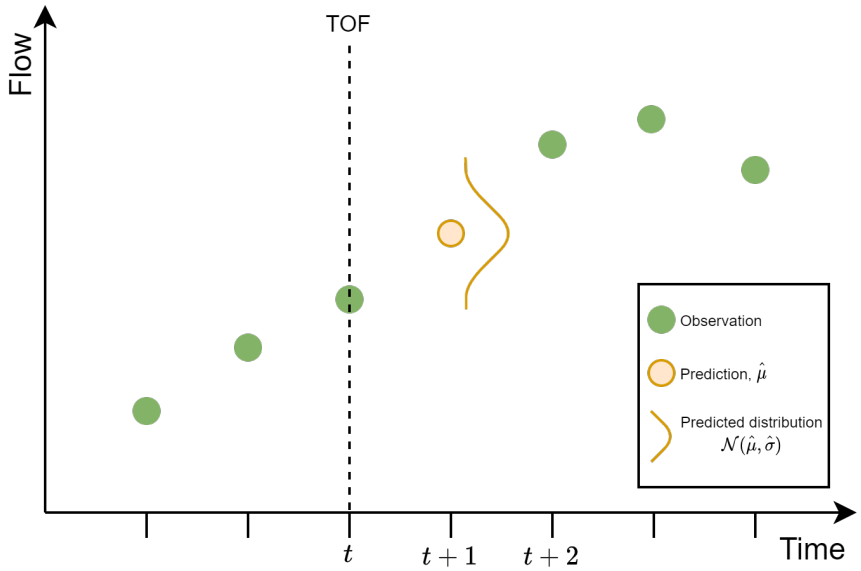


Figure 5.5: Illustration of data imputation using the ANN. At TOF t a forecast is generated for the subsequent time step $t+1$ and the mean of the predicted distribution is used to fill out the missing observation.

Evaluating the data imputation

The data imputation was evaluated both visually and quantitatively. The visual evaluation consisted of comparing the predicted flow against the observed flow. This evaluation was done for a dry weather period and two periods with rainfall, one with low precipitation and low flow and one with high precipitation and high flow. If the model manages to predict the flow correctly, it shows that the model is capable of imputing observations in the given scenario.

The data imputation mechanism was also tested quantitatively by evaluating it in the same way as the models without data imputation. Here, all missing data in the test set was imputed, and the evaluation scores were calculated on the whole dataset (see Section 5.3). By imputing the missing data, forecasts were made at all time steps, which might increase the ability of the model to forecast ATS-activations. Another relevant test of the data imputation would be removing valid data for multiple periods, replacing it with model predictions, and then measuring the goodness of fit. However, this was out of scope for this thesis, as it is only a proof-of-concept model.

5.2.3 Anomaly detection

Discovering anomalies is extremely important when moving towards a more data-driven urban water management, but doing so manually is a tedious and time-consuming task, especially with an increased number of sensors. Therefore, a model-driven anomaly detection approach would be preferred. For this, the predictive distribution from the probabilistic model output can be used. However, its quality must be evaluated first, and an anomaly threshold must be chosen.

Evaluating probabilistic quality from quantiles

It is essential to assert that the probabilistic quality of the model complies with the needed accuracy which can be tested in different ways. In this thesis, it was tested against the theoretical quantiles corresponding to 1, 2, and 3 standard deviations of the normal distribution. The test is carried out by counting the number of observations inside the selected standard deviations. If 1 standard deviation is used, then an observation has a 66% probability of being within this range, hence for many data points, 66% of the observations should fall within the range, if the underlying distribution is in fact normal. For 2 and 3 standard deviations, the probability is 95% and 99.7%, respectively. The quality of the predictive distribution can thus be evaluated against these theoretical quantiles. The test process is depicted in Figure 5.6. The points inside a selected quantile are counted to assess the quality of the distribution. In the figure, the counting results in the actual quantiles 4/7, 5/7, and 6/7 or 57%, 71% and 86%, using 1, 2, and 3 standard deviations, respectively.

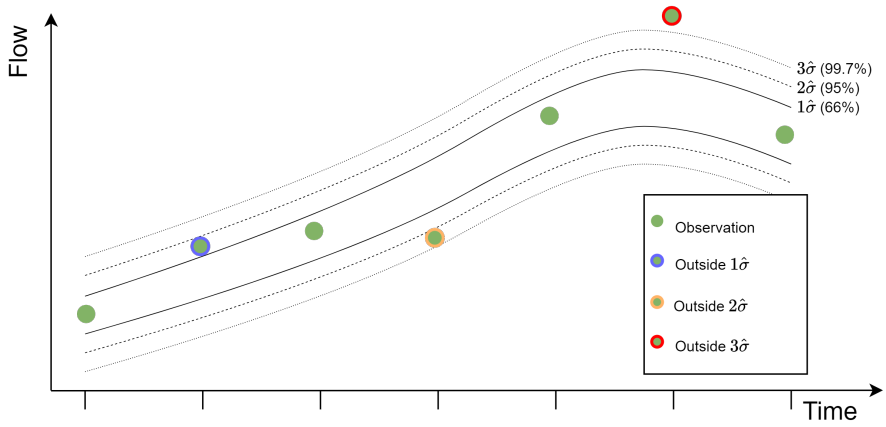


Figure 5.6: Illustration of the quantile evaluation. The quantiles used are the 1, 2, and 3 standard deviations, drawn with solid, dashed and dotted line styles, respectively. The observations are marked with a color if they are outside of a given quantile.

Choosing anomaly threshold

The output distribution from the probabilistic model can be utilized to determine if an observation is an anomaly since the distribution shows the likelihood of an observation being at some point in the distribution. An outlier threshold T was used to determine if the observation was an anomaly. This threshold is a distance related to the characteristics of the distribution, i.e., the standard deviation or a quantile. The observation would be marked as an anomaly if the distance, D, between the predicted mean, $\hat{\mu}$, and the observation was larger than T. This is also illustrated in Figure 5.7 and defined in Equation (5.3).

$$\text{Anomaly} = \begin{cases} \text{True} & \text{if } D > T \\ \text{False} & \text{otherwise} \end{cases} \tag{5.3}$$

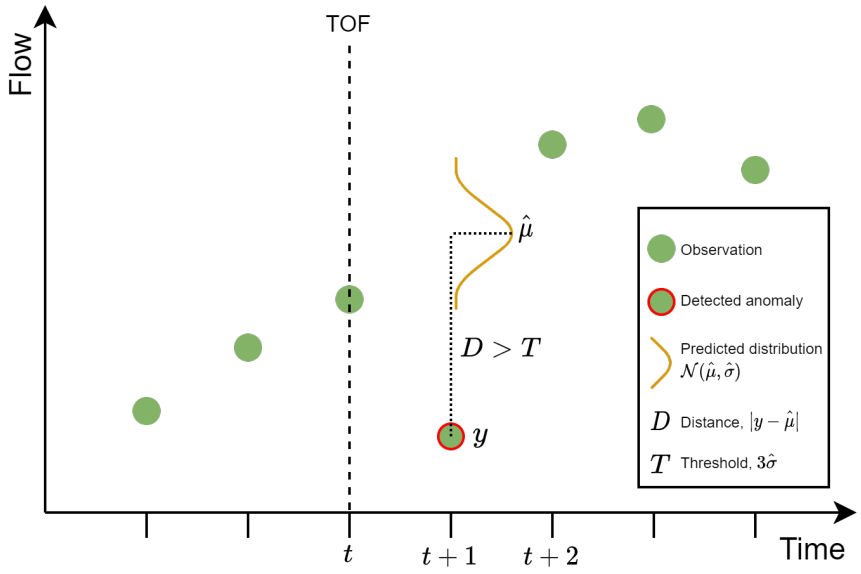


Figure 5.7: Illustration of anomaly detection using the ANN. At TOF t a forecast is generated for the subsequent time step $t+1$. If the distance D between the predicted mean $\hat{\mu}$ and the observation y is larger than a threshold T is marked as an outlier.

where $D = |y - \hat{\mu}|$ is the absolute distance between the predicted mean and observation, and T is the outlier threshold. The threshold T has to be chosen by the user and can be based on the quality of the predicted distribution. For example if $T = 3\hat{\sigma}$ this, in theory, corresponds to the observation having a 99.7% probability of being inside the predicted distribution. $\hat{\mu}$ and $\hat{\sigma}$ comes from the predicted normal distribution $\mathcal{N}(\hat{\mu}, \hat{\sigma})$.

5.3 Evaluation metrics

Three evaluation metrics are used to evaluate the models. First, to test the predictive skill of the forecast against the latest observation, the smoothed persistence index (SPI) is used. Here, the latest observation is a mixture of the observed value and an exponential smoother, which is used due to noise in the measurements. Second, the continuous ranked probability score (CRPS) is used to evaluate the quality of the forecast since this score can take into account both point predictions and probabilistic forecasts. Finally, the critical success index (CSI) evaluates the model in a classification setting.

5.3.1 Smoothed Persistence Index

The SPI is adapted from Löwe et al. (2016) and is similar to the persistence index (PI) (Bennett et al., 2013). The SPI is used due to noise in the flow data and measures the predictive skill of the model relative to a benchmark created using exponential smoothing.

Exponential smoothing is one of the simplest forms of time series analysis models where the model predictions are an exponential weighting of the past observations. The prediction is given by Equation (5.4) (Hyndman and Athanasopoulos, 2018).

$$\hat{Y}_{t+1|t} = \alpha Y_t + \alpha(1-\alpha)Y_{t-1} + \alpha(1-\alpha)^2 Y_{t-2} + \ldots + \alpha(1-\alpha)^T Y_{t-T} \tag{5.4}$$

where $\hat{Y}_{t+1|t}$ is the prediction at timestep $t+1$, Y_t is the observation at timestep t, T is the number of previous timesteps to use and α is the smoothing factor. To get the best performance of the exponential smoothing, the parameter α needs to be tuned to minimize the sum of squared errors (SSE) shown in Equation (5.5).

$$SSE = \sum_{t}^{T} e_t^2 \tag{5.5}$$

where e_t is the error between the observation Y_t and the prediction $\hat{Y}_t$ at time t. To minimize the objective in Equation (5.5), the L-BFGS-B bounded gradient descent algorithm (Zhu et al., 1997) is used to search for the optimal α, where $\alpha \in [0, 1]$.

Based on the exponential smoother, the equation for the SPI is shown below in Equation (5.6).

$$SPI = 1 - \frac{\sum_{t=1}^{T} \left(\sum_{h=1}^{H} \left| Y_{t+h} - \hat{Y}_{t+h} \right| \right)^2}{\sum_{t=1}^{T} \left(\sum_{h=1}^{H} |Y_{t+h} - ((1-\lambda) \cdot Y_{SM,t-1} + \lambda \cdot Y_t)| \right)^2} \tag{5.6}$$

where T is the number of time steps, H is the length of the forecast horizon, Y_{t+h} is the observation at time step t and horizon h, $\hat{Y}_{t+h}$ is the prediction at time step t and horizon h, $Y_{SM,t-1}$ is the exponentially smoothed prediction at the last time step, and λ is the weighting parameter between the last observation and the exponentially smoothed prediction. The weighting parameter is found by minimization of the denominator based on the training data using the L-BFGS-B algorithm with a constraint on the boundary for $\lambda \in [0, 1]$.

The SPI is calculated as an average across all forecast horizons and time steps on the test set. Like the original PI, a perfect forecast across all forecast horizons will have the value of 1, whereas 0 indicates that the forecast is no better than using a smoothed version of the most recent observation.

5.3.2 Continuous Ranked Probability Score

To evaluate the probabilistic forecasts the CRPS first introduced by Epstein (1969) is used. The CRPS has previously been used in atmospheric science where forecasts comes with a probability given by ensembles (Bröcker, 2012; Candille and Talagrand, 2005; Hersbach, 2000), but also in environmental science with probabilistic forecasting of overflow volumes using grey-box models (Löwe et al., 2014, 2016).

The CRPS is sensitive to distance, meaning that larger credit is given for higher probabilities, closer to the real target, which was shown by Gneiting and Raftery (2007). Deterministic forecasts correspond to placing 100% probability at a single point. Thus CRPS can be used to compare deterministic and probabilistic forecasts. Hence a point measure corresponds to finding the absolute error (AE). The CRPS is shown in Equation (5.7) and corresponds to taking the integral over all brier scores at threshold j for the continuous predictive distribution (Brier, 1950; Gneiting et al., 2005).

$$CRPS(F, y) = \int_{-\infty}^{\infty} [F(j) - \mathcal{H}(j - y)]^2 \, \mathrm{d}j \tag{5.7}$$

$$\mathcal{H}(j - y) = \begin{cases} 0 \text{ if } j < y \\ 1 \text{ if } j \geq y \end{cases} \tag{5.8}$$

where F is the predictive normal CDF for j, y is the observation, $\mathcal{H}(j - y)$ is the Heaviside function (Equation (5.8)), which is equal to 1 if $j \geq y$. The integral can be solved numerically by inserting a range of values for j, or by the closed form solution found in Equation (5.9) (Gneiting and Raftery, 2007).

$$CRPS\left[\mathcal{N}\left(\mu, \sigma^2\right), y\right] = \sigma \left[\frac{1}{\sqrt{\pi}} - 2\phi\left(\frac{y-\mu}{\sigma}\right) - \frac{y-\mu}{\sigma}\left(2\Phi\left(\frac{y-\mu}{\sigma}\right) - 1\right)\right] \tag{5.9}$$

where $\phi\left(\frac{y-\mu}{\sigma}\right)$ and $\Phi\left(\frac{y-\mu}{\sigma}\right)$ denotes the PDF and CDF, respectively. The normal distributions are with mean 0 and variance 1 and are evaluated at the normalized prediction error, $(y - \mu)/\sigma$. The closed form solution was used in cases with no $\log_1 0$-transforms for computational speed.

The average over a given forecast horizon was found by the mean of CRPSs calculated for each forecast horizon, and the equation can be formulated as:

$$CRPS_h = \frac{1}{T} \sum_{t=1}^{T} CRPS\left(F_{t+h}, y_{t+h}\right) \tag{5.10}$$

where T is the number of timesteps and F_{t+h} and y_{t+h} is the predictive CDF and observation, respectively, at time step t for forecast h.

The CRPS was calculated for forecast horizons of 10, 30, 60, 120, and 180 minutes and was averaged across time steps for each forecast horizon. Furthermore, it was calculated on the test set. Interpreting the CRPS can be difficult and is only done to compare different models, where lower is better, and 0 is a perfect score. However, a score of zero is unrealistic since this would imply that the model puts all probability onto a single point.

5.3.3 Critical Success Index

The CSI evaluates a model in a binary classification setting. Since the flow is a continuous variable (in discrete time steps), the prediction must be binarized from a threshold defining the beginning of an ATS-event. Based on the model predicting the occurrence of ATS, the predictions can be split up into different categories called true positives (TPs), false positives (FPs), and false negatives (FNs) in a confusion matrix as defined in Courdent et al. (2017) and Jóhannesson et al. (2021). To correctly classify the ATS-event at time $t + h$, it needs to be predicted within a time window of 60 minutes before ATS-activation to 15 minutes after, and the categories are defined below:

TP: Predicted flow is correctly estimated as being above a threshold.

FP: Predicted flow is incorrectly estimated as being above the threshold, while the observed flow is below the threshold.

FN: Predicted flow is incorrectly estimated as being below the threshold, while the observed flow is above the threshold.

The case where the flow is correctly estimated as being below the threshold, is classified as a true negative (TN), but is not interesting in this case since no actions inside the WWTP are made during dry weather conditions. The CSI is calculated based on the TP, FN and FP in Equation (5.11) and estimates the model's capability of classifying events being above the threshold at which the WWTP operators should take action. This forecasting procedure is also shown in Figure 3.2. The CSI is calculated on an event-basis by assuming that a period of 4 hours needs to pass without any precipitation before a new ATS-event can begin to avoid cases where the ATS is triggered consecutively.

$$CSI = \frac{TP}{TP + FN + FP} \tag{5.11}$$

The threshold at which ATS is activated is set at 6400 m^3/h as defined by the operator of the WWTP.

The CSI is calculated for the same forecast horizons as the CRPS: 10, 30, 60, 120, 180 minutes and is only calculated on an event-basis in the test set. This means that only one TP, FN, or FP is generated at each rain event. A perfect classification would give a CSI of 1 meaning that all ATS-events are correctly classified and the lowest score is 0 meaning that no ATS-events are correctly classified.

6 Results

In the following sections, the results of the methods are presented. First, a preliminary data analysis will show the correlation structures and data distributions. This is followed by the flow forecasting models, where the model combinations with probabilistic or non-probabilistic output, transformations, sample weights, and included rainfall forecasts are compared against each other. Finally, the probabilistic model is tested in other cases to explore how it imputes missing data over extended periods during dry and wet weather and detects anomalies.

6.1 Preliminary Analysis

This section analyses the correlations in the catchment between rain intensity and flow, and presents the distribution of the flow data.

6.1.1 Correlations in catchment

To test how long a hindsight is needed to explain the flow based on the rainfall intensity, the individual rain gauges and an average of these are lagged in 10-minute time steps against the flow observations. The cross-correlation is calculated using the cross correlation function (CCF) at the WWTP inlet and shown in Figure 6.1. The placement of the rain gauges can be found in Figure 3.1.

As seen in the figure, the correlation tops around lags 8-16 on the different rain gauges with cross-correlations of approximately 0.4-0.5. The peak of the correlation is very defined for some rain gauges, e.g., 5771 or 5775 at lag 10-11, whereas rain gauges 5697, 5698, and 5699 are flatter with the peak around lag 12-18. The number of lags correlates with the distance between the rain gauges and the WWTP inflow as expected. From the figure, it is seen that the average rain intensity has a higher peak correlation with the inflow to the WWTP of approximately 0.6 at lag 12. Hence, this feature is used instead of the individual rain gauge time series. The lower number of rainfall intensity inputs will reduce computation time but could lower the performance since the model cannot learn correlations between the individual rainfall intensities. Based on the peak of the lags, the look-back period should at least include lags up until 12. However, a look-back period of 4 hours is chosen to be on the safe side, corresponding to 24 lags.

6.1.2 Data distributions

Figure 6.2 shows how the data is distributed. Most of the observations are between $0\,m^3/s$ and $2.0\,m^3/s$, centered around $1.0\,m^3/s$ which corresponds to the dry-weather flows. A small amount of measurements are also seen between $2.0\,m^3/s$ and $3.0\,m^3/s$, which most likely are rain events with low rain intensity. This small range is followed by a long tail corresponding to increasingly more extreme events, with the extreme events having a flow of $7.0\,m^3/s$. The same trend is eventually shown in the plotted empirical CDF where approximately 90% of the data is below $2.0\,m^3/s$. This implies that only 10% of the data covers 70% of the observed flow range, making it increasingly more difficult to

predict extremes. However, as we will see, the model is fairly good at generalizing to new observations.

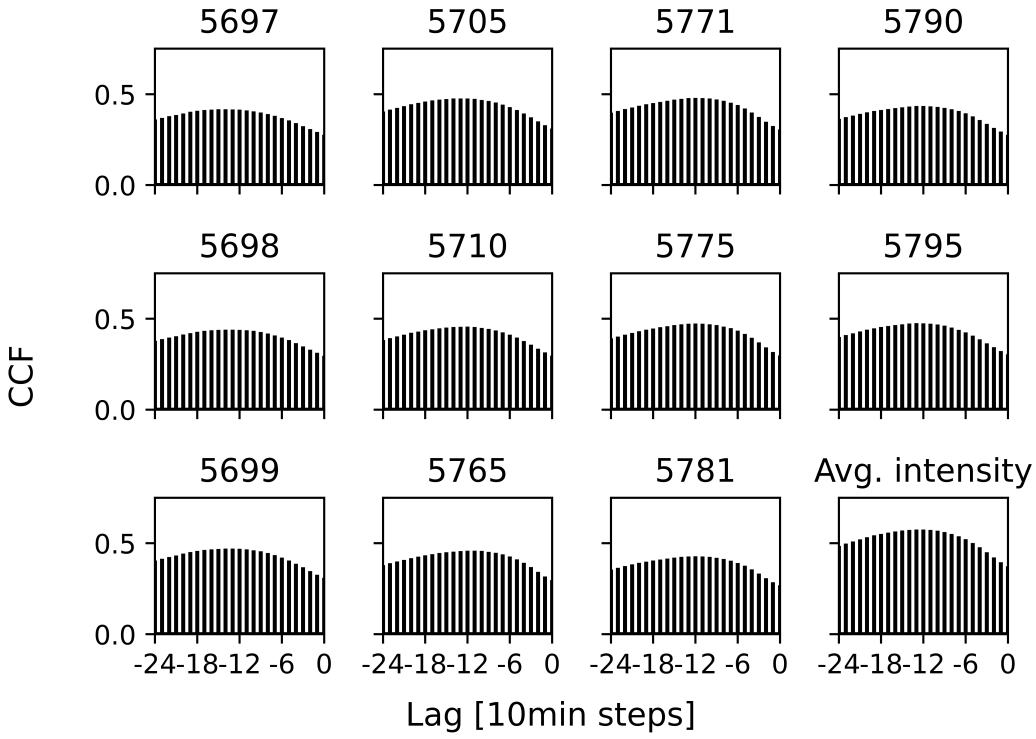


Figure 6.1: Cross correlation (calculated using the cross correlation function (CCF)) between flow at the WWTP inlet and rain gauges and the averaged rain intensity in the system. Rain gauges are lagged in 10 minute steps compared to the flow observations.

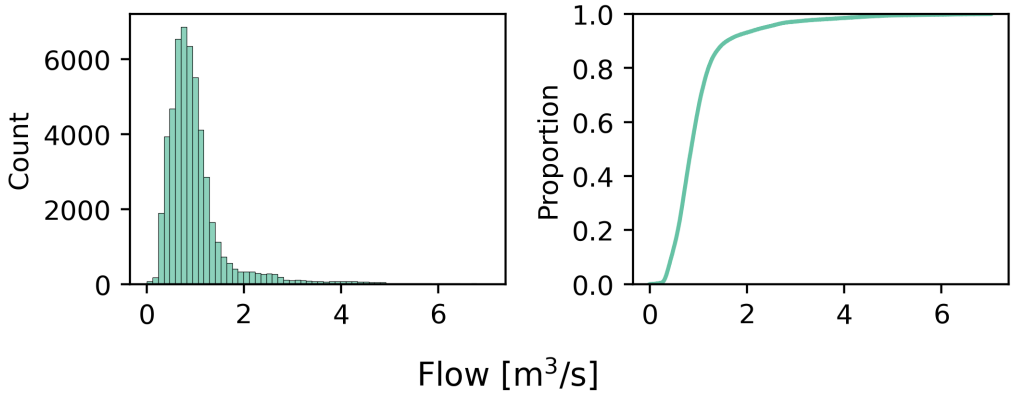


Figure 6.2: The distribution of the number of flow measurements at different flow levels of the test dataset both shown as a histogram and as an empirical CDF.

6.2 Flow forecasting models

This section contains the results of the trained flow forecasting models. First, a summary of the metrics is shown in Section 6.2.1, followed by a visual inspection of the models in

Section 6.2.2 and finally, the results are summarized with main takeaways in Section 6.2.3.

6.2.1 Summary metrics

Five models for each of the initial model combinations, shown in Table 5.2, are trained on the training data, and the SPI, CRPS and CSI are calculated on the test data with the average of the five model performances being shown in Table 6.1. The CRPS and CSI are evaluated with a forecast horizon of 10, 30, 60, 120, and 180 minutes.

Table 6.1: Evaluation metrics for the initial model combinations. For each combination five models are trained and the average of the score of the models are used. The CRPS and CSI are evaluated with a lead time of 10, 30, 60, 120, and 180 minutes indicated by the "_". Green is better.

	SPI	CPRS_10	CPRS_30	CPRS_60	CPRS_120	CPRS_180
FFNN	0.358	0.195	0.200	0.190	0.237	0.278
FFNN-P	0.328	0.131	0.140	0.141	0.179	0.222
FFNN-PT	-0.876	0.345	0.343	0.342	0.353	0.367
FFNN-PW	0.340	0.129	0.138	0.139	0.177	0.221
FFNN-PTW	-0.732	0.326	0.325	0.325	0.338	0.355
FFNN-P-RF	0.336	0.141	0.149	0.149	0.163	0.174
LSTM	0.566	0.152	0.167	0.167	0.215	0.257
LSTM-P	0.585	0.103	0.119	0.128	0.168	0.210
LSTM-PT	0.553	0.145	0.163	0.172	0.212	0.251
LSTM-PW	0.581	0.104	0.120	0.128	0.170	0.213
LSTM-PTW	0.554	0.146	0.163	0.172	0.212	0.250
LSTM-P-RF	0.687	0.104	0.121	0.128	0.146	0.156

	CSI_10	CSI_30	CSI_60	CSI_120	CSI_180
FFNN	0.615	0.576	0.403	0.060	0.048
FFNN-P	0.442	0.374	0.338	0.042	0.027
FFNN-PT	0.266	0.195	0.026	0.000	0.002
FFNN-PW	0.421	0.346	0.396	0.045	0.029
FFNN-PTW	0.310	0.230	0.038	0.000	0.002
FFNN-P-RF	0.431	0.283	0.216	0.188	0.057
LSTM	0.714	0.556	0.503	0.182	0.094
LSTM-P	0.785	0.554	0.506	0.169	0.093
LSTM-PT	0.666	0.501	0.443	0.135	0.084
LSTM-PW	0.792	0.558	0.519	0.177	0.103
LSTM-PTW	0.682	0.500	0.441	0.138	0.085
LSTM-P-RF	0.751	0.521	0.504	0.452	0.411

From Table 6.1, the LSTM architectures are generally seen to be superior for the problem at hand overall. They have the lowest SPI while being able to better extrapolate and forecast ATS-events correctly for longer horizons resulting in better CSI-scores. Furthermore, they also have more narrow uncertainties allowing for lower CRPSs when compared to similar FFNN architectures. When comparing the non-probabilistic models with the prob-

abilistic models, a better overall estimate of the prediction can be achieved with the probabilistic models when comparing the CRPSs of the two, which is true for both the FFNN and the LSTM models. For the FFNN, the CSI is substantially worse for the probabilistic, which could be due to convergence errors, with the learning rate being too high or too low. However, investigating this would require a full hyper parameter search and is out of scope for this thesis, but is discussed in Section 7.3.1.

Moreover, $\log_1 0$ transforming the data did not increase the model's performance, rather the opposite, with both the FFNN and the LSTM models being substantially worse than without the transformation. The same can be seen when having both sample weights and transformations of the data, although the sample weights seem to increase the performance slightly in this combination. Even though sample weights do not increase the general performance of the SPI and CRPS, they do have a slightly positive effect on the CSI. This increase could be linked to its predictive capabilities being weighted more toward predicting the peak events.

For all LSTM models, the forecast performance substantially drops from the 60-minute to the 120-minute horizon. This drop can be alleviated by including the forecasted rainfall. If this is included, the SPI, CRPS, and CSI with forecast horizons of 120 and 180 minutes improves substantially for the LSTM. In contrast, the forecast horizons of 10, 30, and 60 minutes stay unchanged or worsen. This show that reliable forecasts can be made within the catchment response time, where rainfall forecasts should be included if longer horizons are needed. The decrease in performance can be due to the increased number of inputs. The number of parameters stays the same, and the LSTM model improves while the FFNN model does not improve when the rainfall forecasts are included. The missing improvement of the FFNN is likely due to the increased number of inputs relative to the number of neurons in the model, which prevents the FFNN from identifying relevant trends. Detailed confusion matrices for this analysis are included in Appendix D to support these conclusions.

6.2.2 Visual model inspection

Four forecasts are made at different times during an event for different model types in Figure 6.3 to visually compare the different model combinations and see in which cases the models perform well and where they could be improved. A single event is chosen based on its features, having both a dry period and a heavy rain event of a short duration leading to a spike in the flow rate.

The effects of training a model using the negative log-likelihood (NLL) (the maximum likelihood approach) compared to the mean squared error (MSE) can be seen in the first panel of the figure. The models have similar prediction patterns for all scenarios except during the event where the LSTM-P are more smooth, indicating that the probabilistic training also tends to change the mean of the forecast by smoothing it. Both models also fail to predict the increase in flow since they only contain data until the time of forecast (TOF). Since the forecasts between the models are similar, the model optimized with the NLL offers a considerable advantage by having uncertainties of the predictions estimated.

When comparing the LSTM to the FFNN architecture in the second panel in Figure 6.3, it

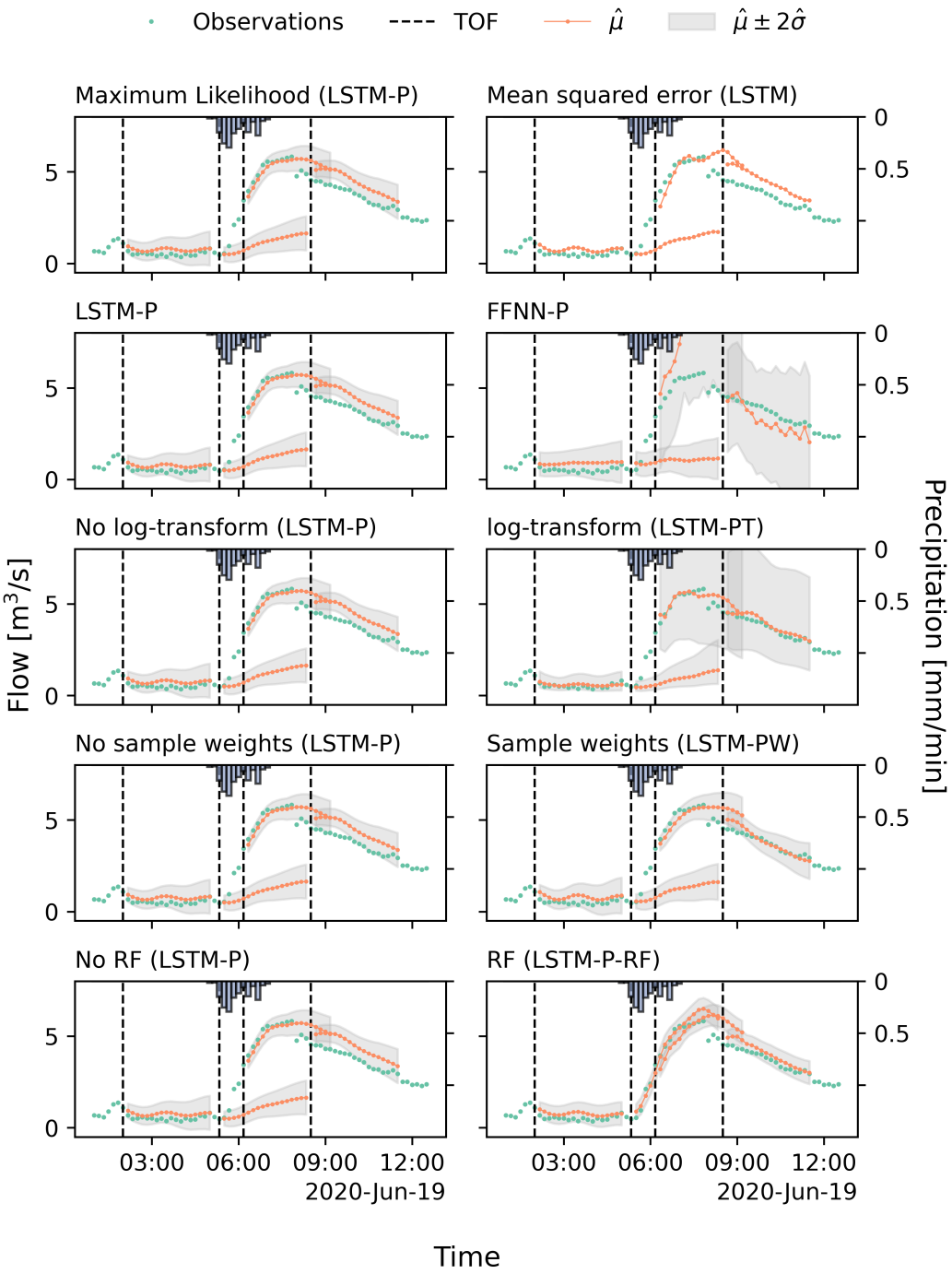


Figure 6.3: Model forecasts for the inflow to Damhusåen WWTP during a rain event using the different combinations shown in Table 5.2. The first panel compares different loss functions during training with a maximum likelihood approach (left) and uses the MSE (right). The second panel compares the use of LSTM (left) against the use of FFNN (right). The third panel compares data transformations using raw data (left) and log-transformed data (right). The fourth panel compares the use of sample weights during training (right) and not using sample weights (left). The fifth panel compares using available data until the time of forecast (left) and also using a perfect rainfall forecast (RF)(right). The event and time of forecast are shared across the figures.

becomes clear that using the model structure build for sequenced data should be favored instead of the FFNN. This supports the conclusion from Table 6.1. The FFNN model captures dry-weather periods and the period after the event, albeit the predictions are linear and highly uncertain in the respective scenarios. Nevertheless, it fails at the start of the event (most likely due to not having forecast data) and during the event, where the model greatly overestimates the peak flow. Furthermore, the uncertainty of the predictions during high flow explodes, which is also reflected in the larger CRPS in Table 6.1. It is also clear that the ANNs can assign a larger uncertainty during rain events than during dry periods.

A $\log_1 0$-transformation was also performed on the data to test whether a data transformation could increase the model's performance. The result is shown in the third panel of Figure 6.3. The model has similar predictions to the LSTM-P, although, a higher uncertainty can be seen during high flow events due to the transformation back into normal space. The data transformation back to normal space does not look like an overall better characterization of the forecast uncertainty confirmed by the CRPS-scores in Table 6.1. Furthermore, the predictions are not as smooth during the high flows.

The model was optimized using sample weights to weigh more infrequent observations, e.g., peak flows, higher in the optimization. The result can be seen in the fourth panel of Figure 6.3. The models with and without sample weights have very similar predictions. In this case, it also better predicts the time after the event, but this could be different for other events or trainings. Having sample weights also slightly increases the forecast uncertainty as a function of the forecast horizon.

The last configuration includes rainfall forecast into the data the model receives, shown in the fifth panel in Figure 6.3. Again, the predictions are similar, but the model can now predict the beginning of the event. Furthermore, the forecast during the event also gets a correct bend following the actual flow. When forecasting the dry-weather flow and the flow after the event, the prediction intervals are also reduced by adding forecasted rain. This reduction is likely because the model now has information about the future, thus not considering the uncertainty for possible precipitation.

6.2.3 Summary

Based on the comparison of the different models, the LSTM-P is the obvious choice since it correctly captures time dependencies and contains simple inputs. If forecasts are available, they should be used to increase predictive performance outside of or close to forecast horizons longer than the catchment response times because they can utilize the information given by the rainfall forecast.

Furthermore, not having to work with transformed data and sample weights is preferred since it adds extra complications and no considerable improvements followed with these initiatives. The model also showed to have the highest SPI and lower CRPSs for forecast horizons greater than 60 minutes than similar models like LSTM-P and LSTM-PW. However, it should be noted that the LSTM-PW did have a higher CSI when compared to the similar models; hence training the data with weights might increase the ATS-event prediction. The key takeaways are presented in bullets below:

- Non-probabilistic (MSE) and probabilistic (maximum likelihood) models yields similar forecasts. Using the probabilistic approach to quantify uncertainty estimates included in the model predictions gives a major advantage.
- $\log_1 0$-transformation of the data did not improve the model, but rather the opposite.
- Using sample weights slightly improved the model's performance but also increased the complexity of the setup (selecting the number of bins or different weighting methods) and is not straightforward to implement in a training setup.
- Reliable forecasts can be made up until the catchment response time without including rainfall forecasts. Although depending on the spatio-temporal distribution, the rainfall might appear closer to the inlet, leading to shorter response times and hence a need for rainfall forecasts.

A limitation of the current setup is that observations for the past four hours are needed to make a forecast. The model could fill in the missing observations for historical analyses and in an operational setting to solve this problem. Furthermore, anomalies such as spikes have a high impact on the forecast quality, and the model can likely be used to detect and replace these to obtain better forecasts. This is researched below in Section 6.3.

6.3 Model-based data assessment

The model has been trained to forecast inflow to the WWTP for the Damhusåen catchment. In this section, however, the usage of the model is tested for different applications. First, in Section 6.3.1, it is tested if the model can be used to impute missing data for dry and wet conditions and how it improves the model's performance. Second, in Section 6.3.2, the model is used to find anomalies in the data. Finally, key takeaways are summarized in Section 6.3.3

6.3.1 Data imputation

Filling in missing data is essential when making forecasts in an operational setting since missing observations can lead to faulty or no forecasts. Therefore, the first part of this section covers a visual inspection of how the model behaves when using its predictions as inputs to making new forecasts. The second part evaluates the forecasts generated across the whole test dataset, with model imputed values. Finally, for illustrative purposes, it is shown how the model can be used to impute missing data for an extended period with missing data.

Visual inspection

A visual inspection is carried out only based on rain intensity and time encodings, which is the data available when sensor flow data is not, to assert how well the model can predict the flow. Three different periods are selected from the test set, and the observations are removed. Removing whole days and letting the model fill in missing values is an extreme case, and most of the time, the input will lack a maximum of a couple of measurements. The model is then used to predict the missing observations using its predictions as inputs to the next prediction as described in Section 5.2.2, where the flow data have been removed for the selected periods. The three periods can be seen in Figure 6.4 where

the data used for making predictions is gradually replaced by predictions when the time increases.

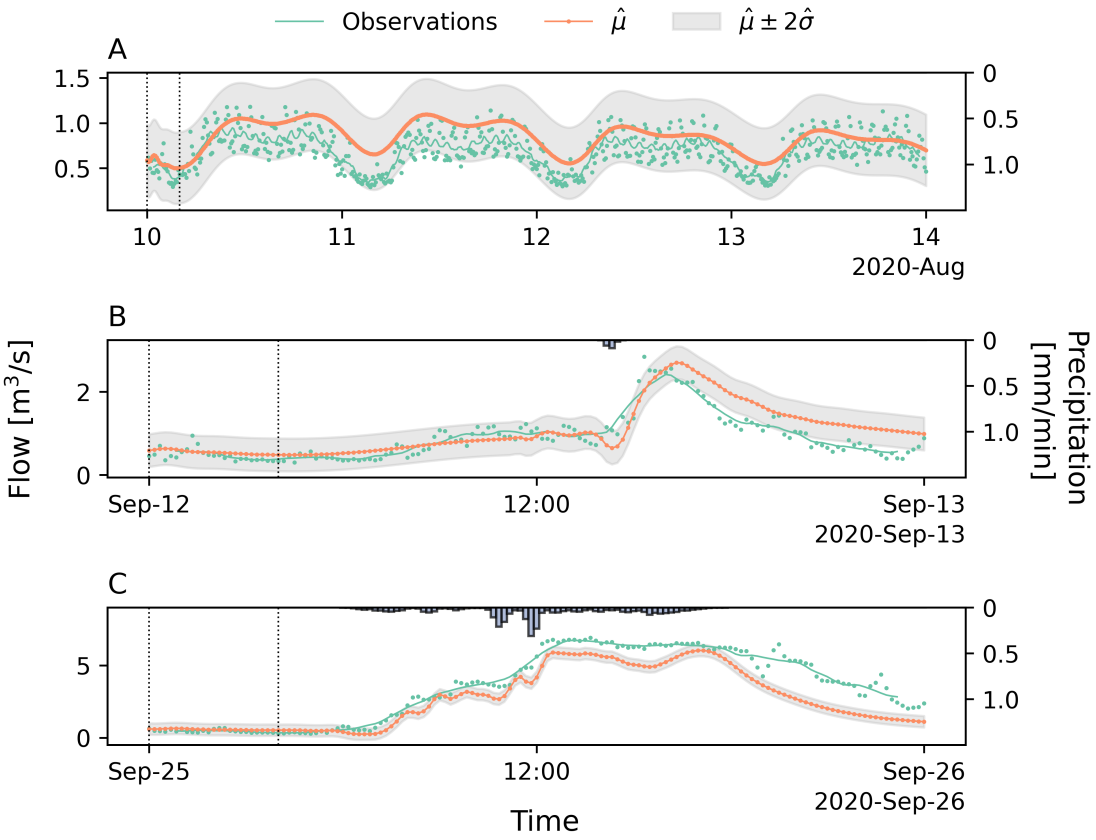


Figure 6.4: Data imputation using the LSTM-P-RF model. The observations are shown as a centered 2-hour rolling average and dots to show the noise of the observations. The predictions are shown with the mean $\hat{\mu}$ and the prediction uncertainty of $2\hat{\sigma}$. Vertical black dots show the period from having only measured data (first line) and gradually using more predicted data to having only predicted data (second line). Panel A shows the imputation of a dry-weather flow in August. Panel B is the imputation of a low flow rain event in September. Panel C is the imputation of a high flow rain event in September.

The model is used for imputing data for four days from the 10th to the 14th of August 2020, which is a period with dry weather. The imputation is shown in panel A in Figure 6.4. Based on this, the model slightly overestimates the dry weather flow during peak and non-peak flows. However, almost all observations are within the predicted two standard deviations. This overestimation could be due to lower base flows in the summer compared to other seasons. Since the model is not trained with an annual time encoding, it will not be able to differentiate between the seasons. Furthermore, the variations in the flow seem to decrease over time. However, when looking at an extended period (Appendix E), the model has just learned weekly flow variations, meaning that the training data suggests that the flow is higher on Mondays and Tuesdays and lower in weekends. The model performance on data imputation could likely be improved by including the time of year or an indicator of the last known base flow as a feature. Closer to the first black line, the predicted flow is noisier since observations are still the driving factor of the prediction. In

contrast, closer to the second black line, the prediction is smoother because predicted values are now the main driver. The same can be seen for the other panels.

During wet conditions, the model might behave differently, and thus two different scenarios are chosen to test it. The first scenario is a period with low rain intensity from the 12th to the 13th of September 2020, and the second scenario is an event with high rain intensity from the 25th to the 26th of September 2020. The imputation for the two events is shown in panels B and C in Figure 6.4, respectively.

As seen in panel B, the model can correctly time but overestimate the magnitude of the peak flow and continues to overestimate the flow after the peak. From 12:00 until the beginning of the event, some irregularities, where the flow suddenly starts to fluctuate a little, followed by a drop in the flow, can be seen. This period corresponds to when the model starts getting information about the forecasted precipitation and is not what is expected. It could be something that is learned if control measures reduce the flow slightly at the beginning of an event. Moreover, although it is not as expected, the predictions and the predicted uncertainties are still within a reasonable range compared to the measured noise. When testing the model imputation performance on a high rain intensity event in panel C, the timing of the events is correctly determined. However, the magnitude of the peak flow is underestimated by approximately $0.5\,\mathrm{m}^3/\mathrm{s}$ to $1.5\,\mathrm{m}^3/\mathrm{s}$. Furthermore, the flow continues to be underestimated after the event when the flow decreases. The underestimation is most likely due to the emptying of basins or delays upstream in the catchment, which the model has no information about.

The uncertainty is the same in all panels, even when the model uses its predictions, and is a flaw of the current setup since the model cannot consider the uncertainty of its predictions. In Section 7.3, ways to alleviate this are discussed. Based on these evaluations, the model could be used to fill in missing data and thereby indicate the flow levels in such periods. However, the user should take the predictions with caution since differences between the predicted and the observed values in some places are substantial, as shown by the analysis above.

Imputing real missing observations

The test set contained ten days from the 19th to the 29th of August 2020, where no observations were available and multiple rain events occurred. For illustrative purposes, the model has been used to impute the missing data for the ten days in Figure 6.5.

Based on this, it was found that the ATS would have been activated 5-6 times during the ten days according to the modeled flow. Based on earlier imputation performance and visible correlations with the precipitation, the ATS have most likely been activated on multiple occasions. Although only the 10-minute prediction is shown in the figure, a full 3-hour forecast is generated at each time step, which could have been used to notify the operators of possible incoming ATS-events, even when the sensors were malfunctioning.

Summary metrics when imputing missing data

It was found that missing data affected the model's capability of accurately predicting the beginning of ATS-events in cases where data was missing before an event. By imputing all the missing data in the whole test dataset with the mean, as shown in Figure 5.4,

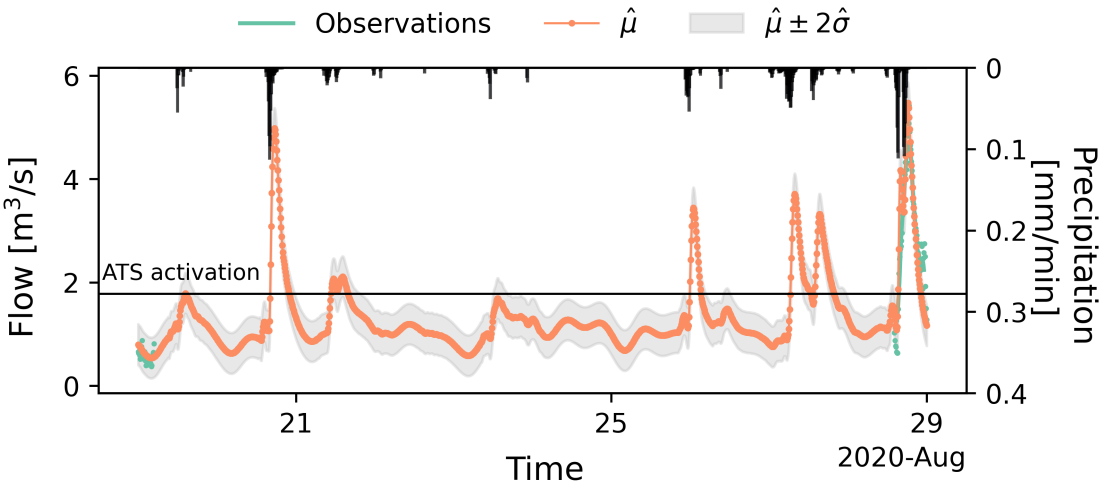


Figure 6.5: Data imputation of real missing data using the LSTM-P-RF model. The few observations available before and after the missing data are shown as dots and as a 2-hour rolling average. The predictions are shown with the mean, $\hat{\mu}$, and the prediction uncertainty of $2\hat{\sigma}$. The flow at which the ATS is activated is shown with the black line. An enlarged version of the figure can be found in Appendix E.

the performance of the model can be calculated again for both SPI, CRPS and CSI. The results are shown in Table 6.2 and show how the model would have performed given it could impute missing data.

Table 6.2: Evaluation metrics for the LSTM-P-RF model with and without imputed missing data computed on the entire test dataset. The CRPS and CSI are evaluated with a forecast horizon of 10, 30, 60, 120 and 180 minutes indicated by the "_". Green is better.

	SPI	CPRS_10	CPRS_30	CPRS_60	CPRS_120	CPRS_180
No imputation	0.687	0.104	0.121	0.128	0.146	0.156
Imputation	0.680	0.104	0.121	0.129	0.147	0.156

	CSI_10	CSI_30	CSI_60	CSI_120	CSI_180
No imputation	0.751	0.521	0.504	0.452	0.411
Imputation	0.800	0.521	0.462	0.492	0.424

The SPI drops a little, while the CRPS remains unchanged when compared to the same model in Table 6.1. These changes are because the model's predictions are used, which is inferior to using the actual observations when making new predictions. At the same time, the uncertainties do not change since the model cannot consider the added uncertainty of its predictions. However, the CSI increases in three of the five calculated horizons (10, 120, and 180-minute), decreases in one (60-minute), and one has no change (30-minute). Although this could change with more events, it seems that using the model to fill in data increases the overall ability of the model to predict ATS-events.

Based on the above results, the model can be used to impute missing data and make flow forecasts to inform operators about possible future ATS-events and to improve existing predictions since the model will fail to make forecasts if data is missing.

6.3.2 Anomaly detection

Detection of anomalies, also referred to as novelties, abnormalities, etc., is also a hot topic with the increased data streams (see Blázquez-García et al. (2021) for a review of published methods). From that review, Munir et al. (2019), Y. Zhang et al. (2012), Zhou et al. (2018), and Zhou et al. (2016) are parametric model-based methods of varying complexity and can in some way be compared with the current setup. With the probabilistic methods presented in this thesis, the anomaly detection can be based on distances from the predicted distribution as outlined in Section 5.2.3.

Evaluation of probabilistic forecast accuracy for one-step predictions

The one-step predictions are shown in Figure 6.6 with varying levels of uncertainty to evaluate the probabilistic forecast accuracy for one-step predictions.

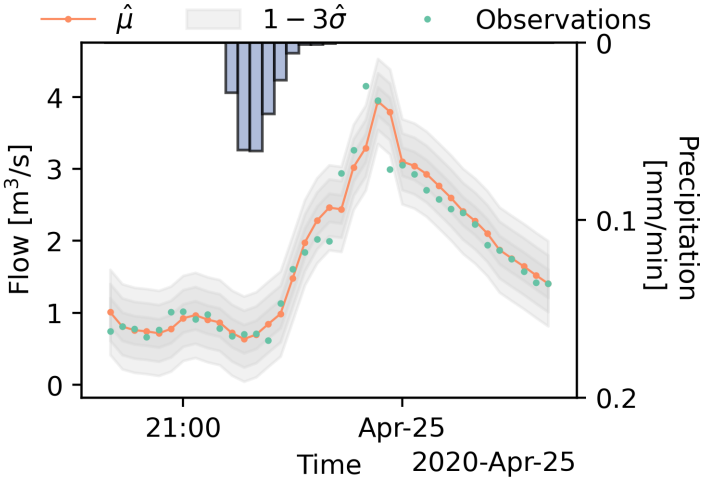


Figure 6.6: Example of the LSTM-P-RF model predictions $\hat{\mu}$ and three degrees of uncertainty described by the predicted standard deviations $\hat{\sigma}$.

Table 6.3: Percentage of observations inside the uncertainty band at 1, 2 and 3 predicted standard deviations $\hat{\sigma}$ from the predicted mean $\hat{\mu}$. Both the expected and actual percentage is shown.

	% of observations inside uncertainty band	
	Expected	Actual
$1\hat{\sigma}$	66.0	88.2
$2\hat{\sigma}$	95.0	98.0
$3\hat{\sigma}$	99.7	99.5

The comparison reveals that most observations are within the inner-most band, and very few fall outside the outer-most band. It also illustrates that the model can estimate the likelihood of an observation being *correct*. Albeit, this requires the predicted distribution to be comparable with the underlying distribution. One way to evaluate the uncertainty of the model predictions is to count the observations within the uncertainty bands at different levels and compare them with the theoretical values as described in Section 5.2.3. The result is shown in Table 6.3, and it can be seen that the empirical and theoretical values do not add up. There are several reasons why the distribution of anomalies is not as according to theory. First, the model predictions are not normally distributed when comparing wet and dry flow regimes. Furthermore, in Figure 6.4, the observations seem to be

distributed with more data points below the mean, meaning that the data will be skewed. However, it is seen that the outer-most band does capture the data quite well when looking at the Figure 6.6 and the comparison of the theoretical and actual values using three standard deviations.

More generally, the observations lying outside the outer-most band are found in high-flow regimes. This could imply that the model does not consider the higher uncertainties and added noise during high flows and needs a different distribution or architecture to account for these errors.

Visual performance of anomaly detection

The model can be used to mark anomalies based on a selected threshold. Based on the above results, the threshold of 3 standard deviations is closest to the theoretical values. Furthermore, anomalies are likely not considered a large part of the data. Therefore targeting the 0.5% outside of the uncertainty band could seem like a good choice. Figure 6.7 shows the anomalies marked by the model for approximately one day. From the figure, it can be seen that most of the anomalies are found in a period after midnight with large fluctuations, which seems valid. Another group of observations marked as anomalies by the model is following a period with rainfall, where the model predicts that the flow should be higher. Finally, a couple of anomalies are found when rapid changes in the flow are present from one time step to another, which on many occasions are also valid.

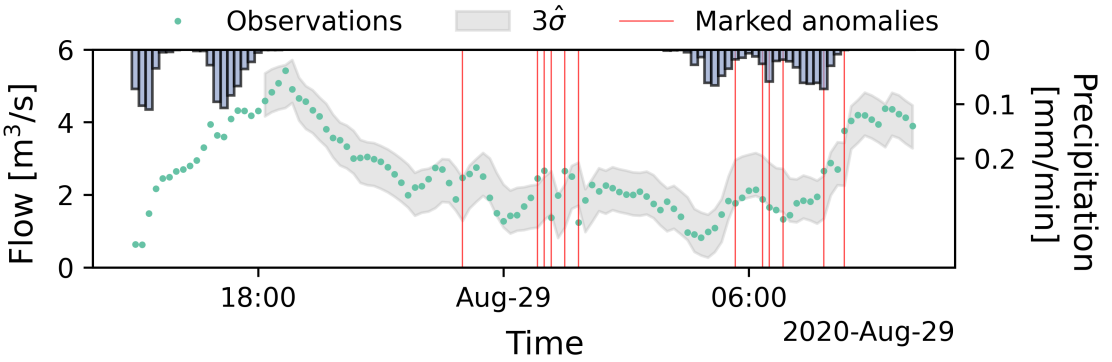


Figure 6.7: The LSTM-P-RF model has been used to mark anomalies in a selected period with rainfall. The uncertainty band starts at 18:00 since observations are missing before the beginning of the event, making the model unable to make forecasts.

Based on these results, the model might be able to detect anomalies. However, before concluding this, the model will have to be tested on other periods where known anomalies are present. This could, for example, be the flatline or spike in Figures 4.3 and 4.4, or other periods with less obvious errors annotated by an expert of the system.

Visual performance of combined anomaly detection and imputation scheme

Assuming that the ANN can model the underlying distribution to the degree that is acceptable for the operator, it is likely that the model could be used to clean the data series by replacing detected anomalies (observations outside a given uncertainty band) with one of the imputation methods described in Section 5.2.2. The predicted mean $\hat{\mu}$ is used to replace the anomaly in Figure 6.8, where the replaced anomaly becomes a part of the data used to generate the next forecast (similar to the method described in Figure 5.4).

The replacement is made to show how a cleaned data series would appear depending on the threshold used. The same event as in Figure 6.6 are chosen, and based on the predicted $\hat{\sigma}$-threshold used, the number of replaced anomalies can vary a lot.

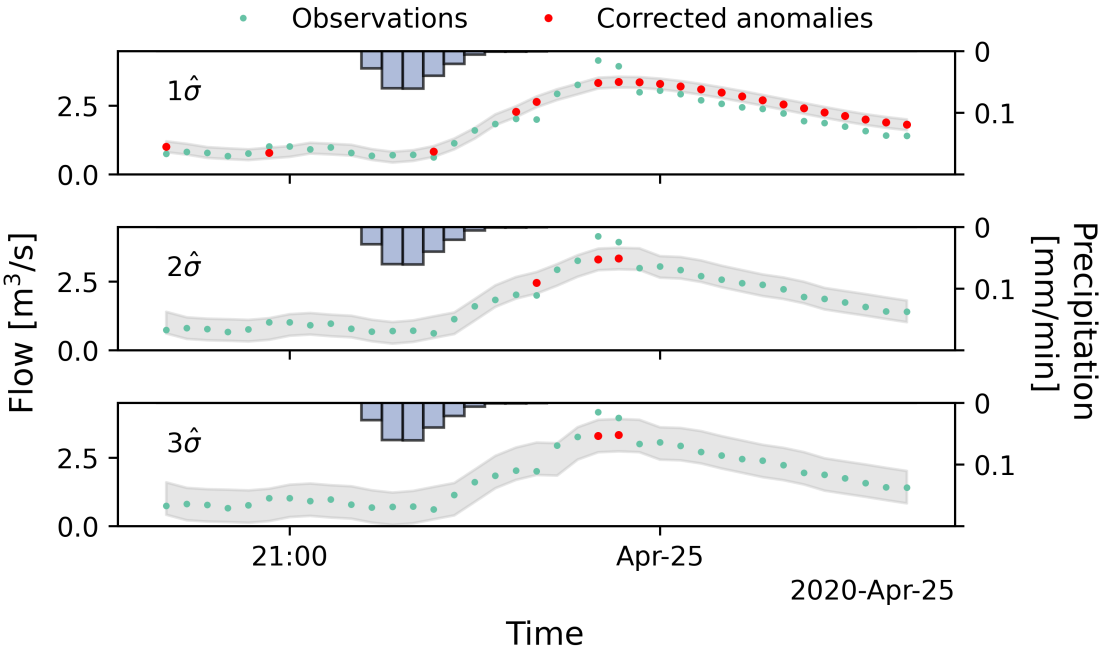


Figure 6.8: Replaced anomalies based on model predictions for a rain event. The first shows the replaced anomalies using threshold $T = 1\hat{\sigma}$. The second panel shows the replaced anomalies using $T = 2\hat{\sigma}$. The third panel shows the replaced anomalies using $T = 3\hat{\sigma}$.

Using the 1 standard deviation threshold replaces a few observations before the event and when the flow is increasing. However, at a point during the event, it begins to replace all observations, since the predictions and uncertainty level never come close to the measured values. This drifting of the model predictions is a flaw in this current setup, where predictions made by the model have the same weight as measured values. However, it might be beneficial for the model to differ between its predictions and observations and adjust uncertainty based on this, which is further discussed in Section 7.3.3. Increasing the threshold to 2 or 3 standard deviations decrease the number of replaced anomalies, but the peak flow is still marked as an anomaly.

Based on Figure 6.8, one should be careful when using the model for automatically cleaning the data since reduced or increased flow could easily replace the measured flows based on the chosen threshold.

6.3.3 Summary

Based on the above analysis, the key takeaways for data imputation are summarized in the points below:

- The model is stable, even for longer imputation times, leading to:
  - A high degree of reliability when filling missing data using observations during

low and high flows.

  - A high degree of reliability when filling missing data using predictions during low flow.
  - A medium degree of reliability when filling missing data using predictions during high flow.

- Prediction uncertainty does not change when using model predictions due to the current setup.
- Imputing data and using this to make forecasts improves the ability of the model to forecast ATS-events in time.

Below, the key takeaways when using the model for anomaly detection are summarized:

- The normal distribution is likely not the best to describe the underlying distribution, since results in uncertainty bands do not meet the theoretical values.
- Automatically replacing anomalies should be done with caution because this could lead to the model ignoring observations.

7 Discussion

The following section discusses the data, methods, and results. The section starts by discussing the results of the flow forecasting models and how they can be used to fill in missing data and detect anomalies. Then, the limitations of the approach are discussed, followed by a research outlook of what can be done to improve the existing approach.

7.1 Flow forecasting models

In summary, this thesis developed a model architecture that can be used for forecasting flows, predicting missing data, and detecting anomalies in time series. The main results and limitations for the different applications are discussed in the following.

7.1.1 Model architecture

Finding an architecture that could converge and produce reliable probabilistic forecasts was the first big part of the thesis, since no similar setups could be found in the literature for this application on probabilistic and multi-step output. Similar setups could be found for the point-prediction, although only for producing 1-step ahead point-predictions in UDSs (Bailey et al., 2016; Palmitessa et al., 2021; D. Zhang et al., 2018a,b). 1-step probabilistic outputs have also been found in the literature for air quality measurements (Murad et al., 2021) and soil moisture (Fang et al., 2020); however, the model architectures and setups are not disclosed. Based on these approaches, the output was extended to produce a forecast with a horizon of 3 hours in 10-minute steps. This output resulted in the novel architecture with separate pipelines for predicting the mean and standard deviations that greatly surpass the performance of an architecture with a single pipeline estimating both.

7.1.2 Forecasting

Both FFNN and LSTM type models have been tested, and, not surprisingly, the LSTM models are superior when working with time series, which is reflected by the evaluation metrics. The models were also trained in a log-normal space to fit the model to more normal distributed data. However, this turned out to make the model worse. It was not tested if the model might need a different optimization scheme or if it could not predict the output and use the data as-is. The model was also trained using sample weights, showing a slight improvement in CSI. However, it is not trivial to implement in the training framework and requires the user to estimate the sample weights. Lastly, forecast rainfall was added as an input which yielded a model that improved on all evaluation metrics. These improvements were especially seen for the forecast horizons close to, or beyond, the response time in the catchment.

Using the model that provides the probabilistic output, both showed an increase in performance regarding the ability to correctly predict the observations at different horizons, reflected in the CRPS, but also yields the uncertainty of a given prediction directly in the form of a predictive distribution. Getting the uncertainty directly through a distribution saves time in an operational setting. In comparison, most other frameworks rely on

sampling the uncertainty by running the models thousands of times or are defined by the residual distribution in the model under different assumptions.

When comparing the results with Jóhannesson et al. (2021), which used ARIMAX models, a forecast horizon of 90 minutes, different train and test periods, and different rain input among other things; the model implemented in this thesis is comparable or slightly worse with respect to performance without hyperparameter tuning. However, the ANN approach has far more flexibility when adding extra inputs or outputs, which are discussed further in Section 7.3. Also, it is not limited to a fixed uncertainty based on residuals.

7.1.3 Data imputation

Missing data points are often found in sensor time series, and though the reasons can be many, filling these gaps can be crucial. Especially when using the data for modeling purposes, like the current setup, where the model will fail to make forecasts if data is missing in the input.

Suppose the model is used to make forecasts iteratively. In that case, the 1-step ahead predictions could, in theory, be used to impute missing data. This was tested on different flow regimes where the model only had to rely on its predictions as input. Here, it was found to accurately impute dry weather flow, even for an extended period, proving it to be stable over longer time frames. The model could also accurately predict the flow to the acceptable degree for low flow rain events. For extreme events, the model can still time the flow, but generally underestimates it. The reason that the model is stable over more extended time frames could be an outcome of the model being trained to predict multiple forecast horizons simultaneously.

Using the model to impute missing data in an operational setting showed an overall improvement of the model's ability to estimate ATS-events reflected in the CSI. In the tests, the model had to rely on only its predictions as input, which is unlikely in most operational applications, and for this, the model could be used to fill in the few missing data to make a forecast. Furthermore, this would most likely result in forecasts close to having the actual observations.

Another application is when post-processing the data for analysis. Often it is too tedious and time-consuming to fill these gaps. For this, the model could also be used, since it showed good performance, even when not having previous observations.

7.1.4 Anomaly detection

The third tested application of the model is for anomaly detection using the model with probabilistic output. Here, the predicted distribution can act as a probability of a given observation being inside or outside of a selected threshold, i.e., if the observation is outside $\hat{\mu} \pm T\hat{\sigma}$ where T is a scalar determining the threshold, then the observations can be marked as an anomaly. However, the underlying distribution is likely not to resemble a normal distribution, so the theoretical probabilities for observations outside T standard deviations does not hold. Because of this, the operator would have to be careful when selecting the threshold. A discussion on how to better model the underlying distribution is found in Section 7.2.2.

Assuming that the desired threshold was found, the model can also be used to replace detected anomalies with model predictions, but should be carried out with great care, as model predictions become part of the input. Moreover, because the model cannot differentiate between its predictions and observations, this can result in the model beginning to ignore observed values in the worst case. Making the model consider if the input is its predictions or observed values would be a considerable addition and is discussed in Section 7.3. Furthermore, different replacement strategies can be used. An anomaly could be replaced by e.g., the predicted mean, $\hat{\mu}$, as done in this thesis. It could also be sampled from the predictive distribution, $\mathcal{N}(\hat{\mu}, \hat{\sigma})$, as described in Section 5.2.2, which would introduce stochasticity. However, both methods are prone to removing the edge cases that are true observations from the data.

7.1.5 Model from an operational setting

From the operator's point of view in an operational setting, one crucial task is to correctly estimate the beginning of the ATS-event. So far, only the mean of the output distribution has been used to assert whether or not the ATS-threshold has been breached. Since the model outputs a distribution and is not optimized for detecting ATS-events, the optimal variable used to detect ATS-events could be something other than the mean. It could be tested if some quantiles yield better ATS-activations than others. Since most of the wrong classifications are late activations, choosing a quantile higher than the mean of the distribution could lead to earlier activations. It would likely not negatively impact the ATS-activation with respect to the correct activations (true positive) because a correct activation is still considered correct within 60 minutes before the actual activation, which is an extensive time range. However, a too high quantile could lead to correctly estimated non-activations (true negative) turning into wrong activations (false positive), thus making a false alarm.

The optimal threshold could be found by assessing the relative economic value of making a wrong prediction which has been applied to select thresholds for radar extrapolation flow predictions used to activate ATS (Courdent, 2017). Instead of varying the threshold of the radar extrapolation flow prediction, the quantile of the predicted distribution used to activate ATS could be optimized.

Another essential part is to fill in missing values and detect anomalies in the system, such as flatlining sensors or clogging of the pipe. Hence, if these applications work out, they could have broader impacts as this can be used in many other places in the systems where sensor data is available. Also, this is not necessarily limited to UDSs, but can be used more generally in water systems.

7.2 Limitations

The limitations of the current setup are discussed with a focus on new storage structures, data distributions, and the model's output.

7.2.1 Added storage structures and control in Damhusåen

Many storages and delay structures already exist in Damhusåen. However, from January the 3rd 2020 to July the 1st 2020, a new large storage pipe was taken into use, adding

extra complexity to the system. Although impacts could not visually be found in the data, they might have an impact. The tunneling was done in two stages, with the first part being actively used from January 3 and the second on July 1. These changes mean that the model is trained on data not affected by this new large storage pipe and might impact the results, with the model predicting more water towards the WWTP. Furthermore, the controls named ICDAM, emptying this new storage pipe was activated on the 1st of February 2020, with changes in control rules from April the 1st 2020. This control adds an extra layer of uncertainties since the filling and emptying of these storage pipes affect the wet-weather flow during rain events and the dry-weather flow, when emptied. Since this storage structure was not taken into use during the train and validation period, it might affect the model's capability under certain conditions during the test period, where water is temporally stored during rain events.

7.2.2 Data distributions

The extreme data distribution challenges the model. A $\log_1 0$-transformation and sample weights have been applied to accommodate this issue. However, other techniques could also have been used and are described below. The first is to try to make the data more normal distributed, whereas the second is to try to model the output as the extreme distribution.

Input data distribution

Instead of weighting the different samples or transforming the input data, down-sampling observation classes with many exemplars, over-sampling, or creating synthetic data for observation classes with few exemplars could have been applied as a pre-processing step.

Down-sampling removes data, which is not preferred when working with ANNs, especially if down-sampling removes the majority of the data. Here, it is essential to keep as much data as possible since the data input drives the model.

Over-sampling for time-series can be done with a version of the synthetic minority over-sampling technique (SMOTE) algorithm has been created by Torgo et al. (2013). The algorithm has been tested for data-driven models to predict wastewater flow (Karimi et al., 2019).

Creating synthetic data with other methods is also an option. It could be done using calibrated forward models, like MIKE+ or SWMM, or newer data-driven methods like Generative Adversarial Networks (GANs) (Esteban et al., 2017; Yoon et al., 2019), but both are difficult or time-consuming, or both.

All of the described methods run the risk of introducing wrong observations that contributes negatively to the underlying distribution of the data for different reasons (e.g., changes in catchment structures over time), forcing the ANN to worsen.

Model output distribution

As was seen in Figure 6.4, the underlying distribution of the data seems to be more heavily towards the right side, with more points lying towards a lower boundary. Furthermore, in Section 6.3.2, the predicted values outside a given threshold do not follow the theoretical

values, indicating that a normal distribution does not describe the underlying distribution. Hence, instead of forcing the data to be normally distributed, the model output distribution could be changed towards a more extreme distribution, matching the underlying distribution like the exponential or the gamma distributions. However, this requires that the actual underlying distribution is known. One way of solving the issue is by using the sinh-arcsinh distribution, which allows for adjusting *skewness* and *tailweight* of the distribution (Jones and Pewsey, 2009). The downside is that it requires estimating an extra two output parameters. However, it has shown promising results in uncertainty modeling of synthetic climate data (Barnes et al., 2021).

7.3 Research outlook

This section centers around possible research directions. First, additions to the existing model setup are presented. Then, a way to incorporate missing data and make the model account for its predictions is discussed. Furthermore, alternative and newer network types are touched upon. Finally, two possible new directions are presented in the form of developing an algorithm that automatically and continuously updates the model and using other sensors in the catchment to stabilize model predictions which is essential if, e.g., flow data is missing.

7.3.1 Possible model architecture improvements

When discussing the results, the model hyper parameters (learning rate, dropout rate, depth and width of the network, activation functions, etc.) are not optimized. Thus, the performance will probably increase if these are calibrated. The optimization can be done using a search scheme, like grid search or Bayesian optimization (Snoek et al., 2012). Bayesian optimization has recently been used for hyper parameter tuning of an LSTM neural network in an environmental setting to predict a sensor output (Hansen et al., 2022).

7.3.2 Alternative network types

In this thesis, the FFNN and LSTM network types are tested. However, other network types might increase performance. One of these network types is the CNN, which is mostly used for image recognition, but can also be used on sequential data using 1D convolutions.

The other model type is the Transformer (Vaswani et al., 2017), which is used in much newer state-of-the-art deep learning models (Brown et al., 2020; Devlin et al., 2018). This network type works by an attention mechanism, where all tokens (observations) are compared with all other tokens to estimate their relevance. Furthermore, the Transformer offers a map of this attention mechanism which allows for some explainability of a prediction, which is often a reason why black-box models are excluded.

Both the CNNs and the Transformers have the added benefit that they do not have to be trained in sequences like the LSTM and can thus be trained with high efficiency on GPUs or TPUs.

7.3.3 Training with missing data

When the model imputes missing data, some artifacts were seen in Figure 6.4 in the flow pattern, when presented to forecasted rainfall. These artifacts could probably be

avoided if the model is trained on a mix of missing data and observations, which would force the model towards not making considerable changes to current flows. Usually, this can be mitigated by training the network using teacher-forcing (Graves, 2013), where the predictions are fed back into the network when data is missing. More stable training methods also exist, such as scheduled sampling (S. Bengio et al., 2015) and professor-forcing (Lamb et al., 2016). Training the model this way could likely improve anomaly detection and data imputation, since the model needs to account for the uncertainty of previous predictions.

7.3.4 Automated model-updating algorithms

A massive problem with having many sensors is the time it takes to collect, clean, and prepare the data, which is also a weak point of data-driven models that often require cleaned data to function. This process could be avoided to a certain extent if an algorithm could be made that automatically updates the model once data comes in and cleans the old data so that it is ready to use. Figure 7.1 illustrates the process mentioned above, of training a model and updating it with new data every second week.

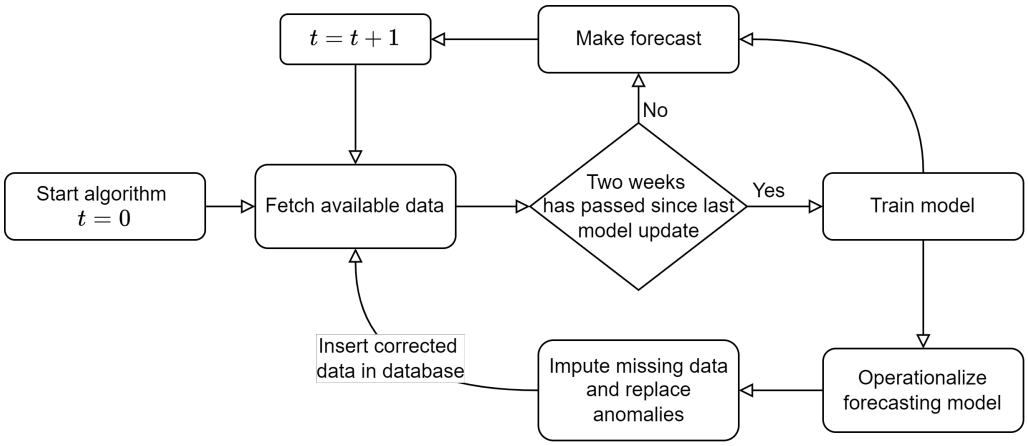


Figure 7.1: Flowchart describing the process of automatically updating a model. In this example, the model is re-trained every two weeks, with the new model being put into production and used for cleaning the existing data. The model running in production will continue to make a forecast for two weeks before being replaced by a newly re-trained model.

The idea is that the model will have large uncertainties in the beginning and probably not be very good at making forecasts. However, over time, the accuracy of the model predictions will increase, and the forecast will be usable. This framework also allows for not having to collect data for many months or years before utilizing them, combined with not having to spend a lot of time re-training the models manually.

The data cleaning process can be omitted or put in a framework where they have to be approved by a user of the system. The substantial uncertainties, in the beginning, lead to the model not replacing observations unless they are extreme, such as huge spikes from sensor malfunctioning. With more data, the model will be better at estimating the flow and subsequently be able to find other anomalies than apparent errors, such as flatlines, small spikes, etc.

The time between the model-retraining likely needs to be dynamic, meaning that in the beginning, it is trained often, and with more data, the time between re-training increases.

7.3.5 Multiple sensors

In this thesis, only the measured inflow to the WWTP was used, although the flow and water levels are measured at multiple places in the catchment as shown in Figure 3.1. With the increased flexibility of the ANNs, these measurements could be added to the input of the model to increase the reliability when filling in missing data and detecting anomalies. These sensor correlations will also increase the model performance, since the sensors are more directly correlated to the inflow of the WWTP. Two applications of the extra sensors exist:

1. **Multiple single networks**: Having multiple networks, estimating only the flow of water level at a certain point, allows the use of predictions from these networks in multi-sensor forecasting networks.
2. **One many-to-many network**: A many-to-many network is a term used for a network with multiple sequences in the input and output. Instead of having a model for each sensor in a network, one could forecast all sensor measurements simultaneously by using all previous sensor measurements in the input and making a forecast of each sensor in the output. Simultaneously simulating the sensors would probably also help further regulate the network predictions and make them less likely to diverge. A many-to-many network has previously been used to model CSOs (D. Zhang et al., 2018a).

On one hand, the advantage of using multiple smaller networks, only predicting one sensor at a time, is that it is easier to work with and offers higher flexibility if one network needs a change. Having a single network, on the other hand, lowers the time spent configuring the individual networks, and it would need to be tested if such a network is feasible.

8 Conclusion

This work aimed to find a proof-of-concept ANN architecture that supports probabilistic forecasts of the inflow to the WWTP and compare this to non-probabilistic forecasts. The predictions have a forecast horizon of 3 hours and a time resolution of 10 minutes and were evaluated using the smoothed persistence index (SPI), continuous ranked probability score (CRPS), critical success index (CSI), and by visual inspection. Furthermore, it was tested if such a model could be used in different contexts, with one filling in missing values and another detecting and replacing anomalies.

When establishing the proof-of-concept model, the following conclusions were drawn:

- The probabilistic model architecture needs separate pipelines for determining the mean and standard deviation of the predictive distribution, whereas only a single pipeline was needed for the non-probabilistic architecture.
- The LSTM models have superior forecast capabilities, when compared to FFNN models, demonstrated by higher SPI and CRPS scores.
- The probabilistic model better characterizes the flow compared to the non-probabilistic model, which is reflected by the CRPS being reduced for all tested forecast horizons for the probabilistic models.
- $\log_1 0$-transforming the flow data decreased the model performance for all evaluation scores. Thus, transforming the data does not lead to any improvements. In contrast, sample weights lead to a slight increase in ATS-detection, reflected by the increase in CSI.
- The forecasting capabilities decrease with an increase in the forecast horizon. However, a substantial decrease was found around the forecast horizon corresponding to the catchment response time. From this, it is deduced that reliable forecasts can be made without rainfall forecasts as input to the model, until the catchment response time. Consequently, if rainfall forecasts are available, this can substantially increase the performance for longer forecast horizons.

Generally, the model was found to have a high capability of imputing missing values. The main findings are expressed below:

- The model yields stable predictions over long time frames when using its predictions as input.
- Data imputation has high reliability during dry weather and during low flow, and medium reliability during high flow. The data imputation was tested in the extremes with no observations present. Therefore, it is likely that the model also has a high reliability during high flows, if previous observations are present and only a few data points need to be imputed.

- Using the model's predictions as input does not change the prediction uncertainty of the next prediction. However, it improves the ability to forecast ATS-activations, shown by a general increase in the CSI, compared to models without data imputation.

More research is necessary when testing the model skill for predicting and replacing anomalies. However, based on the results found, when testing the quality of the predictive distribution and using the model for replacing anomalies, the main findings are presented below:

- To determine if an observation is also an anomaly, it was found that a threshold related to the predictive distribution could be used, which means that all observations outside a given quantile are considered outliers.
- Since the quality of the predictive quantiles did not meet the theoretical quantiles, the detection of anomalies needs to be done with great care, as it can lead to wrong estimates of what are considered anomalies. Furthermore, if anomalies are replaced with predictions, selecting a threshold that is too low might make the model diverge from the actual observations.
- To obtain a more reliable anomaly detection, the predictive distribution needs to match the underlying distribution. Because of this, the normal distribution is likely not optimal. Hence, a distribution that better characterizes the underlying distribution would improve the reliability of anomaly detection.

Overall, it was found that the LSTM models with probabilistic forecasts acquired results similar to what could be found in the literature, although hyper parameters were not optimized. Furthermore, the models can be used to impute missing data with high reliability, except for extreme scenarios with high flow and long time frames with missing data. Additionally, a step toward automatic anomaly detection and replacement has been made, utilizing the probabilistic predictions from the model.

9 Data Management and Programming

During the thesis all data management have been made with Python3 using various libraries `Numpy`, `Pandas`, `Matplotlib`, `SciPy`, `Scikit-Learn`, `TensorFlow2` and `TensorFlow Probability` among others. An exhaustive list as well as versions can be found in the **.yml** file on the github repository at https://github.com/phillipaarestrup/master_thesis or on zenodo with the DOI: 10.5281/zenodo.6726556 or with the URL: https://doi.org/10.5281/zenodo.6726556.

Illustrations for the thesis are made with the free-of-use, open-source software *draw.io* licensed under Apache 2.

Bibliography

Alamia, A., Gauducheau, V., Paisios, D., and VanRullen, R. (2020). “Comparing feed-forward and recurrent neural network architectures with human behavior in artificial grammar learning”. *Scientific reports* 10.1, pp. 1–15.

Arnbjerg-Nielsen, K. (2012). “Quantification of climate change effects on extreme precipitation used for high resolution hydrologic design”. *Urban Water Journal* 9.2, pp. 57–65.

Bailey, J., Harris, E., Keedwell, E., Djordjevic, S., and Kapelan, Z. (2016). “The use of telemetry data for the identification of issues at combined sewer overflows”. *Procedia engineering* 154, pp. 1201–1208.

Barnes, E. A., Barnes, R. J., and Gordillo, N. (2021). “Adding Uncertainty to Neural Network Regression Tasks in the Geosciences”. *arXiv preprint arXiv:2109.07250*.

Bengio, S., Vinyals, O., Jaitly, N., and Shazeer, N. (2015). “Scheduled sampling for sequence prediction with recurrent neural networks”. *Advances in neural information processing systems* 28.

Bengio, Y. (1991). “Artificial Neural Networks and their Allplication to Sequence Recognition”. PhD thesis. McGill University.

Bengio, Y. (2012). “Practical recommendations for gradient-based training of deep architectures”. *Neural networks: Tricks of the trade*. Springer, pp. 437–478.

Bengio, Y., Frasconi, P., and Simard, P. (1993). “The problem of learning long-term dependencies in recurrent networks”. *IEEE international conference on neural networks*. IEEE, pp. 1183–1188.

Bengio, Y., Simard, P., and Frasconi, P. (1994). “Learning long-term dependencies with gradient descent is difficult”. *IEEE transactions on neural networks* 5.2, pp. 157–166.

Bennett, N. D., Croke, B. F., Guariso, G., Guillaume, J. H., Hamilton, S. H., Jakeman, A. J., Marsili-Libelli, S., Newham, L. T., Norton, J. P., Perrin, C., Pierce, S. A., Robson, B., Seppelt, R., Voinov, A. A., Fath, B. D., and Andreassian, V. (2013). “Characterising performance of environmental models”. *Environmental Modelling & Software* 40, pp. 1–20.

Blázquez-García, A., Conde, A., Mori, U., and Lozano, J. A. (2021). “A review on outlier/anomaly detection in time series data”. *ACM Computing Surveys (CSUR)* 54.3, pp. 1–33.

Breinholt, A., Thordarson, F. Ö., Møller, J. K., Grum, M., Mikkelsen, P. S., and Madsen, H. (2011). “Grey-box modelling of flow in sewer systems with state-dependent diffusion”. *Environmetrics* 22.8, pp. 946–961.

Brier, G. W. (1950). “Verification of forecasts expressed in terms of probability”. *Monthly weather review* 78.1, pp. 1–3.

Bröcker, J. (2012). “Evaluating raw ensembles with the continuous ranked probability score”. *Quarterly Journal of the Royal Meteorological Society* 138.667, pp. 1611–1617.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). *Language Models are Few-Shot Learners*. DOI: 10.48550/ARXIV.2005.14165.

Bundgaard, E., Nielsen, M., and Henze, M. (1996). "Process development by full-scale on-line tests and documentation". *Water Science and Technology* 33.1, pp. 281–287.

Candille, G. and Talagrand, O. (2005). "Evaluation of probabilistic prediction systems for a scalar variable". *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography* 131.609, pp. 2131–2150.

Carstensen, J., Nielsen, M. K., and Strandbæk, H. (1998). "Prediction of hydraulic load for urban storm control of a municipal WWT plant". *Water science and technology* 37.12, pp. 363–370.

Courdent, V. (2017). "Numerical Weather Prediction and Relative Economic Value framework to improve Integrated Urban Drainage- Wastewater management". PhD thesis. Technical University of Denmark.

Courdent, V., Grum, M., Munk-Nielsen, T., and Mikkelsen, P. S. (2017). "A gain–loss framework based on ensemble flow forecasts to switch the urban drainage–wastewater system management towards energy optimization during dry periods". *Hydrology and Earth System Sciences* 21.5, pp. 2531–2544. DOI: 10.5194/hess-21-2531-2017.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. DOI: 10.48550/ARXIV.1810.04805.

DHI (2021). *Mike+*. https://www.mikepoweredbydhi.com/products/mikeplus. Online; accessed June 12, 2022.

Dong, L., Fang, D., Wang, X., Wei, W., Damaševičius, R., Scherer, R., and Woźniak, M. (2020). "Prediction of streamflow based on dynamic sliding window LSTM". *Water* 12.11, p. 3032.

Eggimann, S., Mutzner, L., Wani, O., Schneider, M. Y., Spuhler, D., Moy de Vitry, M., Beutler, P., and Maurer, M. (2017). "The potential of knowing more: A review of data-driven urban water management". *Environmental science & technology* 51.5, pp. 2538–2553.

EPA (2020). *Storm Water Management Model (SWMM)*. https://www.epa.gov/water-research/storm-water-management-model-swmm. Online; accessed June 12, 2022.

Epstein, E. S. (1969). "A scoring system for probability forecasts of ranked categories". *Journal of Applied Meteorology (1962-1982)* 8.6, pp. 985–987.

Esteban, C., Hyland, S. L., and Rätsch, G. (2017). "Real-valued (medical) time series generation with recurrent conditional gans". *arXiv preprint arXiv:1706.02633*.

European Commission (2016). *Statistics on cities, towns and suburbs*. DOI: 10.2785/91120.

Fang, K., Kifer, D., Lawson, K., and Shen, C. (2020). “Evaluating the potential and challenges of an uncertainty quantification method for long short-term memory models for soil moisture predictions”. *Water Resources Research* 56.12, e2020WR028095.

Gal, Y. and Ghahramani, Z. (2016). “Dropout as a bayesian approximation: Representing model uncertainty in deep learning”. *international conference on machine learning*. PMLR, pp. 1050–1059.

Glorot, X., Bordes, A., and Bengio, Y. (2011). “Deep sparse rectifier neural networks”. *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, pp. 315–323.

Gneiting, T. and Raftery, A. E. (2007). “Strictly proper scoring rules, prediction, and estimation”. *Journal of the American statistical Association* 102.477, pp. 359–378.

Gneiting, T., Raftery, A. E., Westveld III, A. H., and Goldman, T. (2005). “Calibrated probabilistic forecasting using ensemble model output statistics and minimum CRPS estimation”. *Monthly Weather Review* 133.5, pp. 1098–1118.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. http://www.deeplearningbook.org.

Graves, A. (2013). “Generating sequences with recurrent neural networks”. *arXiv preprint arXiv:1308.0850*.

Hansen, L. D., Stokholm-Bjerregaard, M., and Durdevic, P. (2022). “Modeling phosphorous dynamics in a wastewater treatment process using Bayesian optimized LSTM”. *Computers & Chemical Engineering* 160, p. 107738.

Hersbach, H. (2000). “Decomposition of the continuous ranked probability score for ensemble prediction systems”. *Weather and Forecasting* 15.5, pp. 559–570.

Hochreiter, S. and Schmidhuber, J. (1997). “Long short-term memory”. *Neural computation* 9.8, pp. 1735–1780.

Hyndman, R. J. and Athanasopoulos, G. (2018). *Forecasting: principles and practice*. OTexts.

Jazwinski, A. H. (2007). *Stochastic processes and filtering theory*. Courier Corporation.

Jiang, Y. (2021). “Machine Learning Approaches to Model the Error of Hydrodynamic Drainage Models”. MA thesis. Technical University of Denmark.

Jóhannesson, A., Vezzaro, L., Mikkelsen, P. S., and Löwe, R. (2021). “Approaches for unsupervised identification of data-driven models for flow forecasting in urban drainage systems”. *Journal of Hydroinformatics* 23.6, pp. 1368–1381.

Jones, M. C. and Pewsey, A. (2009). “Sinh-arcsinh distributions”. *Biometrika* 96.4, pp. 761–780.

Jørgensen, H. K., Rosenørn, S., Madsen, H., and Mikkelsen, P. S. (1998). “Quality control of rain data used for urban runoff systems”. *Water Science and Technology* 37.11, pp. 113–120.

Karimi, H. S., Natarajan, B., Ramsey, C. L., Henson, J., Tedder, J. L., and Kemper, E. (2019). “Comparison of learning-based wastewater flow prediction methodologies for smart sewer management”. *Journal of Hydrology* 577, p. 123977.

Kingma, D. P. and Ba, J. (2014). “Adam: A method for stochastic optimization”. *arXiv preprint arXiv:1412.6980*.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). “Imagenet classification with deep convolutional neural networks”. *Advances in neural information processing systems* 25.

Lamb, A. M., Goyal, A., Zhang, Y., Zhang, S., Courville, A. C., and Bengio, Y. (2016). “Professor forcing: A new algorithm for training recurrent networks”. *Advances in neural information processing systems* 29.

Li, D., Marshall, L., Liang, Z., Sharma, A., and Zhou, Y. (2021). “Characterizing distributed hydrological model residual errors using a probabilistic long short-term memory network”. *Journal of Hydrology* 603, p. 126888.

Li, P., Zhang, J., and Krebs, P. (2022). “Prediction of Flow Based on a CNN-LSTM Combined Deep Learning Approach”. *Water* 14.6, p. 993.

Lim, B. and Zohren, S. (2021). “Time-series forecasting with deep learning: a survey”. *Philosophical Transactions of the Royal Society A* 379.2194, p. 20200209.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. (2021). “Swin transformer: Hierarchical vision transformer using shifted windows”. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10012–10022.

Löwe, R., Thorndahl, S., Mikkelsen, P. S., Rasmussen, M. R., and Madsen, H. (2014). “Probabilistic online runoff forecasting for urban catchments using inputs from rain gauges as well as statically and dynamically adjusted weather radar”. *Journal of Hydrology* 512, pp. 397–407.

Löwe, R., Vezzaro, L., Mikkelsen, P. S., Grum, M., and Madsen, H. (2016). “Probabilistic runoff volume forecasting in risk-based optimization for RTC of urban drainage systems”. *Environmental Modelling & Software* 80, pp. 143–158.

Lund, N. S. V., Borup, M., Madsen, H., Mark, O., Arnbjerg-Nielsen, K., and Mikkelsen, P. S. (2019). “Integrated stormwater inflow control for sewers and green structures in urban landscapes”. *Nature Sustainability* 2.11, pp. 1003–1010.

Madsen, H. (2008). *Time series analysis*. Chapman and Hall/CRC.

Makridakis, S., Spiliotis, E., and Assimakopoulos, V. (2018). “Statistical and Machine Learning forecasting methods: Concerns and ways forward”. *PloS one* 13.3, e0194889.

Munir, M., Siddiqui, S. A., Dengel, A., and Ahmed, S. (2019). “DeepAnT: A Deep Learning Approach for Unsupervised Anomaly Detection in Time Series”. *IEEE Access* 7, pp. 1991–2005. DOI: 10.1109/ACCESS.2018.2886457.

Murad, A., Kraemer, F. A., Bach, K., and Taylor, G. (2021). “Probabilistic Deep Learning to Quantify Uncertainty in Air Quality Forecasting”. *Sensors* 21.23, p. 8009.

Nair, V. and Hinton, G. E. (2010). “Rectified linear units improve restricted boltzmann machines”. *Icml*.

Nielsen, M. K., Carstensen, J., and Harremoës, P. (1996). “Combined control of sewer and treatment plant during rainstorm”. *Water Science and technology* 34.3-4, pp. 181–187.

Palmitessa, R., Mikkelsen, P. S., Borup, M., and Law, A. W. (2021). “Soft sensing of water depth in combined sewers using LSTM neural networks with missing observations”. *Journal of Hydro-environment Research* 38, pp. 106–116.

Rothenborg, M. (2022). *Biofos topper trist statistik: Bypass fra renseanlæg blev mere end fordoblet i 2021*. https://pro.ing.dk/watertech/artikel/biofos-topper-trist-statistik-bypass-fra-renseanlaeg-blev-mere-end-fordoblet-i. Online; accessed June 20, 2022.

Sharma, A. K., Guildal, T., Thomsen, H., Mikkelsen, P. S., and Jacobsen, B. (2013). “Aeration tank settling and real time control as a tool to improve the hydraulic capacity and treatment efficiency during wet weather: results from 7 years’ full-scale operational data”. *Water science and technology* 67.10, pp. 2169–2176.

Snoek, J., Larochelle, H., and Adams, R. P. (2012). “Practical bayesian optimization of machine learning algorithms”. *arXiv preprint arXiv:1206.2944*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). “Dropout: a simple way to prevent neural networks from overfitting”. *The journal of machine learning research* 15.1, pp. 1929–1958.

Tang, C., Zhao, Y., Wang, G., Luo, C., Xie, W., and Zeng, W. (2021). *Sparse MLP for Image Recognition: Is Self-Attention Really Necessary?* DOI: 10.48550/ARXIV.2109.05422. URL: https://arxiv.org/abs/2109.05422.

Thaileng, T. (2019). “Comparison of Short-Term Rainfall Forecasts for Prediction of Sewer Flow in Urban Area: A Case Study of Damhusåen Catchment, Copenhagen”. MA thesis. Asian Institute of Technology.

Thordarson, F. Ö., Breinholt, A., Møller, J. K., Mikkelsen, P. S., Grum, M., and Madsen, H. (2012). “Evaluation of probabilistic flow predictions in sewer systems using grey box models and a skill score criterion”. *Stochastic Environmental Research and Risk Assessment* 26.8, pp. 1151–1162.

Torgo, L., Ribeiro, R. P., Pfahringer, B., and Branco, P. (2013). “Smote for regression”. *Portuguese conference on artificial intelligence*. Springer, pp. 378–389.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). “Attention is all you need”. *Advances in neural information processing systems* 30.

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

Werbos, P. J. (1990). “Backpropagation through time: what it does and how to do it”. *Proceedings of the IEEE* 78.10, pp. 1550–1560. DOI: 10.1109/5.58337.

Yoon, J., Jarrett, D., and Van der Schaar, M. (2019). “Time-series generative adversarial networks”. *Advances in Neural Information Processing Systems* 32.

Zhang, D., Lindholm, G., and Ratnaweera, H. (2018a). “DeepCSO: Forecasting of Combined Sewer Overflow at a Citywide Level using Multi-task Deep Learning”. *arXiv preprint arXiv:1811.06368*.

Zhang, D., Martinez, N., Lindholm, G., and Ratnaweera, H. (2018b). “Manage sewer in-line storage control using hydraulic model and recurrent neural network”. *Water resources management* 32.6, pp. 2079–2098.

Zhang, L., Wang, S., and Liu, B. (2018). “Deep learning for sentiment analysis: A survey”. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 8.4, e1253.

Zhang, Y., Hamm, N. A., Meratnia, N., Stein, A., Van De Voort, M., and Havinga, P. J. (2012). “Statistics-based outlier detection for wireless sensor networks”. *International Journal of Geographical Information Science* 26.8, pp. 1373–1392. DOI: 10.1080/13658816.2012.654493.

Zhou, Y., Qin, R., Xu, H., Sadiq, S., and Yu, Y. (2018). “A data quality control method for seafloor observatories: The application of observed time series data in the East China Sea”. *Sensors* 18.8, p. 2628. ISSN: 1424-8220. DOI: 10.3390/s18082628.

Zhou, Y., Arghandeh, R., and Spanos, C. J. (2016). “Online learning of contextual hidden markov models for temporal-spatial data analysis”. *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, pp. 6335–6341. DOI: 10.1109/CDC.2016.7799244.

Zhu, C., Byrd, R. H., Lu, P., and Nocedal, J. (1997). “Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization”. *ACM Transactions on mathematical software (TOMS)* 23.4, pp. 550–560.

A Feed-forward neural network

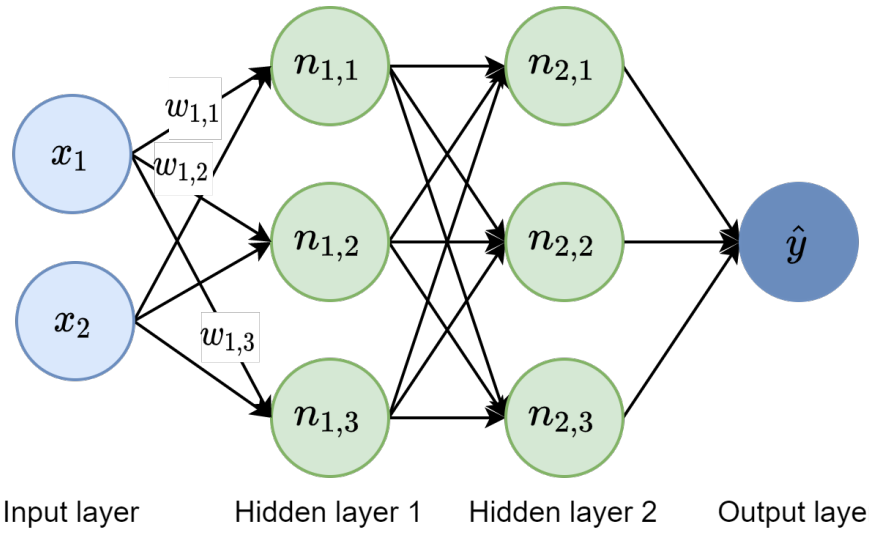
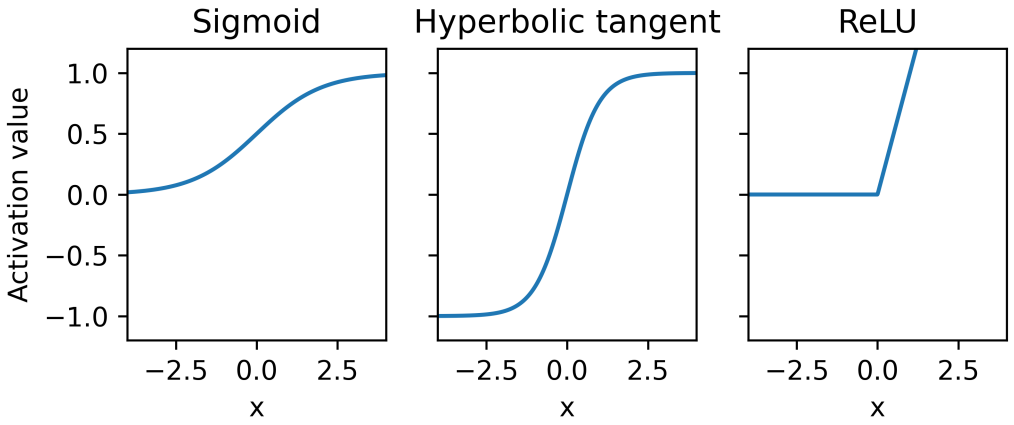


Illustration of FFNN with two inputs, two layers with three nodes and one output. The network takes a number of inputs x, which are connected to the nodes n in the first hidden layer through weights w. These nodes are then connected to each of the nodes in the second hidden layer, which are then connected to the number of outputs y. Notations for the nodes are $n_{\text{layer,node}}$ and for the weights $w_{\text{from,to}}$.

B Activation functions

This appendix contains everything related to the activation functions.

Below, the three activation functions used in this thesis are shown: the sigmoid σ, the hyperbolic tangent tanh and the rectified linear unit (ReLU). The equations for the three activation functions are also shown.

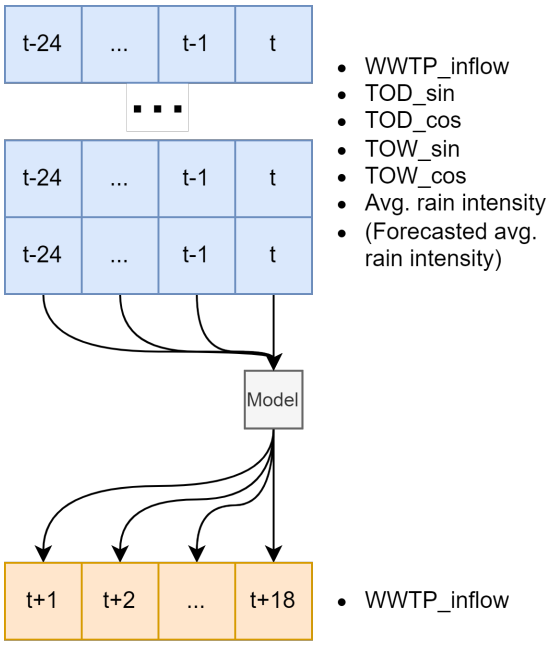


The activation value as a function of the input value for three different activation functions. Left panel is the sigmoid, middle is the hyperbolic tangent and right is the rectified linear unit

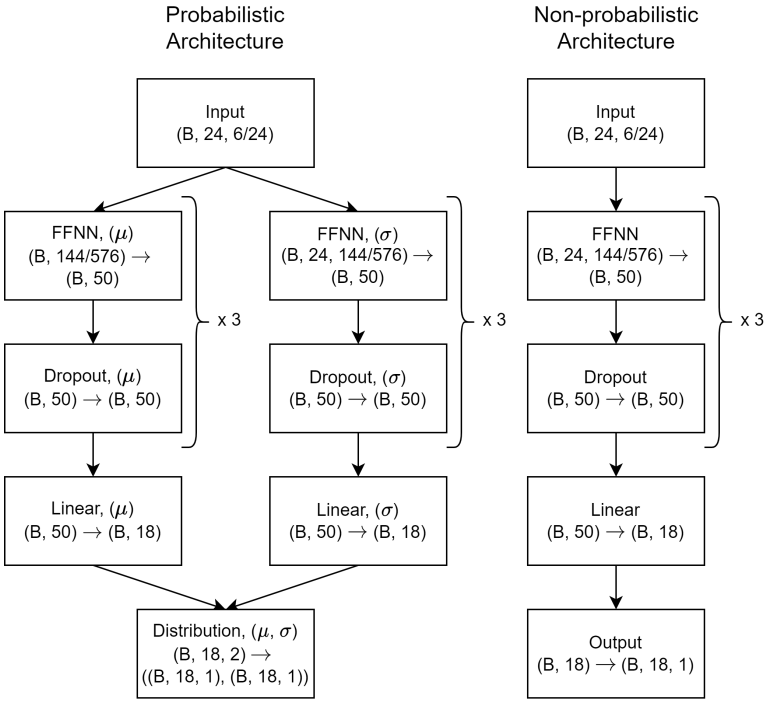
$$\sigma(x) = \frac{1}{1 + e^{-x}}$$
$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$
$$\text{ReLU}(x) = \max(0, x)$$

C Model Architectures

This appendix contains a conceptual overview and the model architectures of the FFNN model.



Overview of the FFNN model. The model receives inputs from the previous $[t-24, \ldots, t-1, t]$ observations and predicts the full output sequence $[t+1, t+2, \ldots, t+18]$ simultaneously. The time series used for the inputs and the predicted output is also shown.

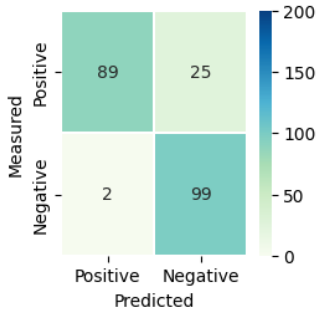


Flow forecasting model architecture for the fully-connected probabilistic and deterministic models. The probabilistic uses two parallel fully-connected layers to predicting μ and σ, respectively, for the normal distribution, whereas, the the deterministic only have a single pipeline predicting the output $\hat{y}$. Reshaping layers are omitted from the figure to reduce complexity but can be found by the shapes going in and out of the layers in the format (input shape) $\rightarrow$ (outputshape) shown below the layer name. For example, (B, 24, 6/24) corresponds to a tensor with dimensions B (batch size), 24 (length of the input sequence), and 6 or 24 (number of inputs with or without rainfall forecast). Furthermore, the FFNN layers are of the same type as Linear but have an ReLU activation applied to make them non-linear. The shapes 144 and 576 appears as the input dimensions are reduced from 3 to 2, meaning that 24x6 and 24x24 becomes 144 and 576, respectively.

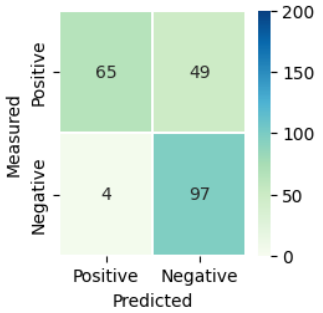
D Confusion matrices, ATS

This appendix contains confusion matrices from the LSTM-P, LSTM-P-RF, and the LSTM-P-RF with data imputation models. The matrices show the true positive (TP), true negative (TN), false positive (FP), and false negative (FN) based on the model correctly predicting the ATS-activation. Only a single model is presented here, so the confusion matrices might slightly differ from another model run.

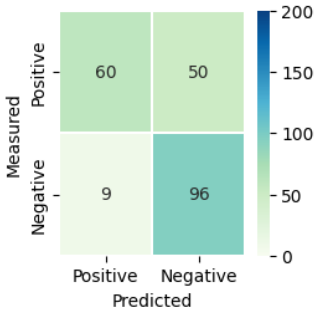
Below the confusion matrices for the LSTM-P model are present.



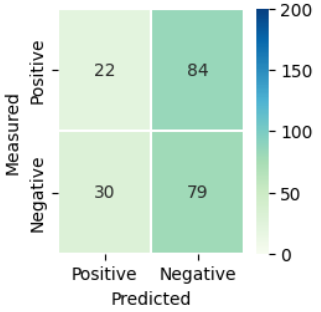
(a) 10 minute forecast horizon



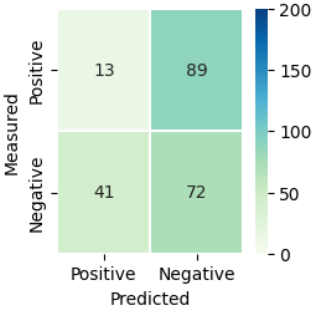
(b) 30 minute forecast horizon



(c) 60 minute forecast horizon



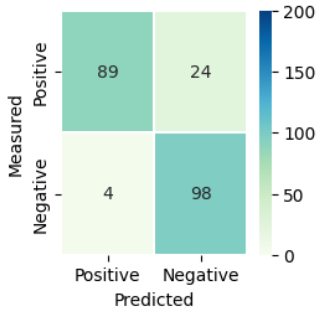
(d) 120 minute forecast horizon



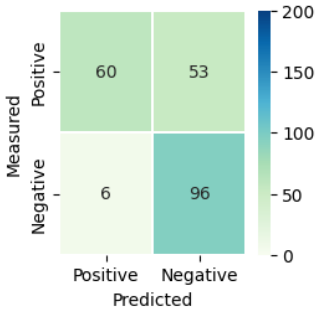
(e) 180 minute forecast horizon

Confusion matrices for the 10, 30, 60, 120 and 180 minute forecast horizons for the LSTM-P model.

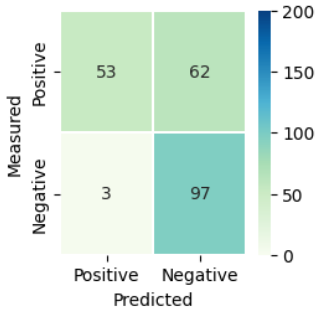
Below the confusion matrices for the LSTM-P-RF model are present.



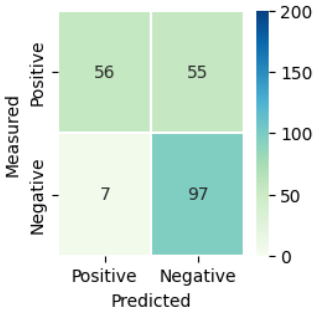
(a) 10 minute forecast horizon



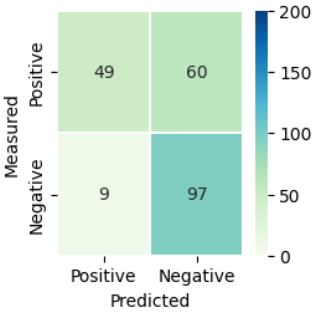
(b) 30 minute forecast horizon



(c) 60 minute forecast horizon



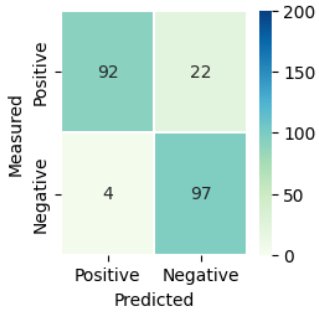
(d) 120 minute forecast horizon



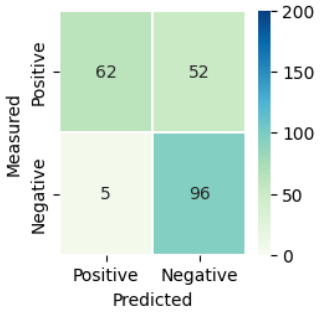
(e) 180 minute forecast horizon

Confusion matrices for the 10, 30, 60, 120 and 180 minute forecast horizons for the LSTM-P-RF model.

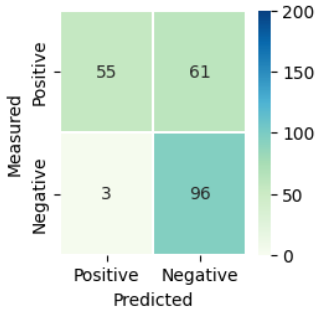
Below the confusion matrices for the LSTM-P-RF with data imputation model are present.



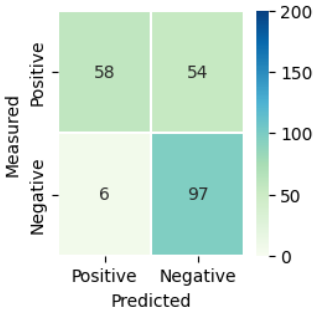
(a) 10 minute forecast horizon



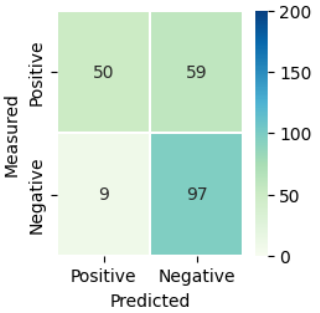
(b) 30 minute forecast horizon



(c) 60 minute forecast horizon



(d) 120 minute forecast horizon

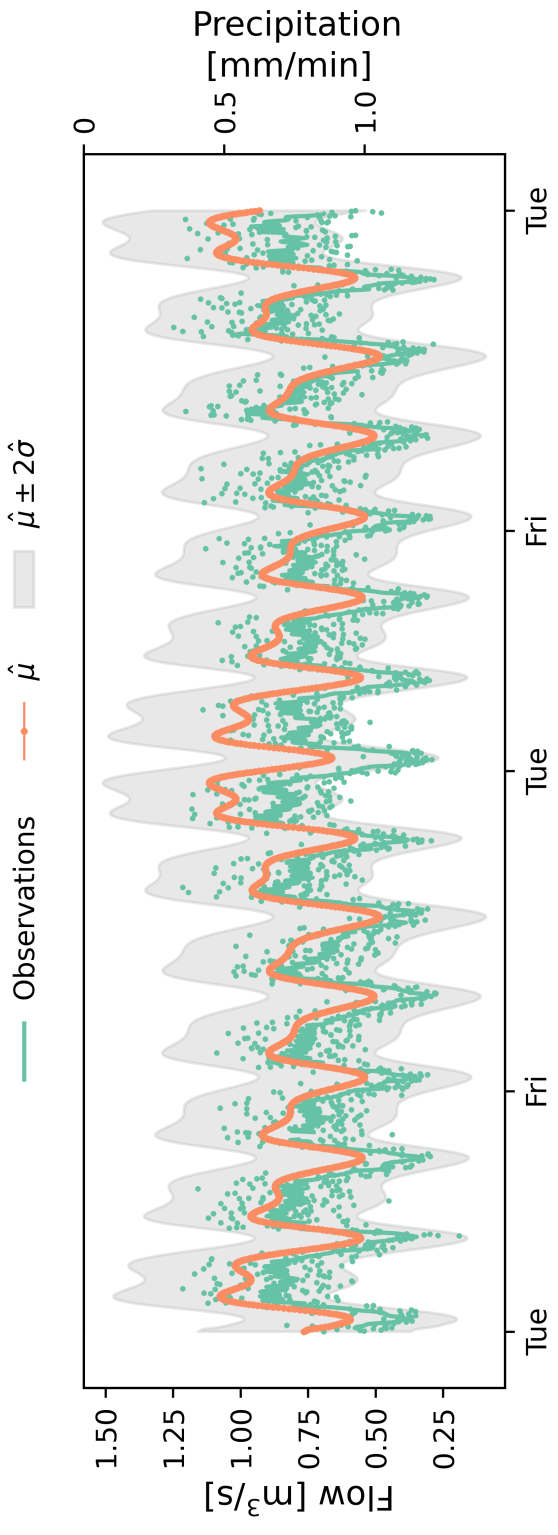


(e) 180 minute forecast horizon

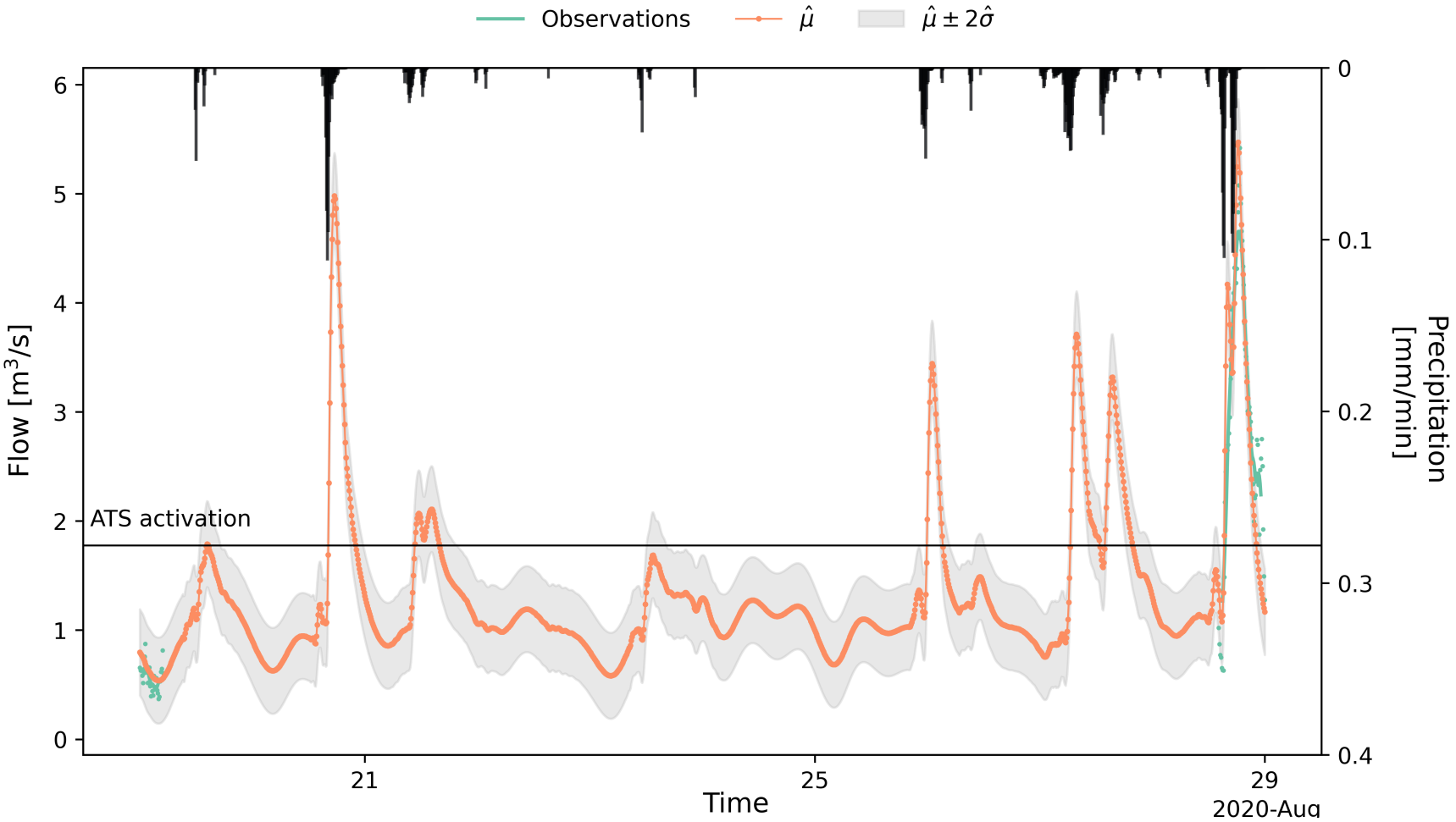
Confusion matrices for the 10, 30, 60, 120 and 180 minute forecast horizons for the LSTM-P-RF model with data imputation.

E Data imputation

This appendix contains everything related to data imputation.



Data imputation during dry weather from the 4th to the 18th of August 2020 using the LSTM-P-RF model. The observations are shown as a centered 2-hour rolling average and dots to show the noise of the observations. The predictions are shown with the mean $\hat{\mu}$ and prediction uncertainties of $2\hat{\sigma}$.



Data imputation of real missing data using the LSTM-P-RF model. The few observations available before and after the missing data are shown as dots and as a 2-hour rolling average. The predictions are shown with the mean $\hat{\mu}$ and prediction uncertainties of two standard deviations $2\hat{\sigma}$. The flow at which the ATS is activated is shown with the black line.

Technical
University of
Denmark

Bygningstorvet, Building 115
2800 Kgs. Lyngby
Tlf. 4525 1700

https://www.sustain.dtu.dk/