ST. PÖLTEN UNIVERSITY
OF APPLIED SCIENCES

Informatik & Security
fh st.pölten

Forecasting Water Flow Rate with Machine Learning

Comparison of a Hydrological and two Machine Learning models to forecast the Water Flow of the Erlauf river in Lower Austria

Diplomarbeit

zur Erlangung des akademischen Grades

Diplom-Ingenieur/in

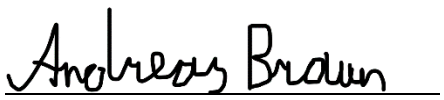
eingereicht von

Andreas Braun, BSc

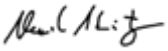
im Rahmen des
Studiengangs Data Intelligence an der Fachhochschule St. Pölten

Betreuung
Betreuer/Betreuerin: FH-Prof. Dr. Alexander Adrowitzer

Wösendorf,
07.09.2023



(Unterschrift Autor/Autorin)



(Unterschrift
Betreuer/Betreuerin)

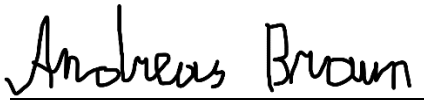
Ehrenwörtliche Erklärung

Ich versichere, dass

- ich diese Diplomarbeit selbständig verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und mich sonst keiner unerlaubten Hilfe bedient habe.
- ich dieses Diplomarbeitsthema bisher weder im Inland noch im Ausland einem Begutachter/einer Begutachterin zur Beurteilung oder in irgendeiner Form als Prüfungsarbeit vorgelegt habe.
- diese Arbeit mit der vom Begutachter/von der Begutachterin beurteilten Arbeit übereinstimmt.

Der Studierende/Absolvent*in räumt der FH St. Pölten das Recht ein, die Diplomarbeit für Lehre- und Forschungstätigkeiten zu verwenden und damit zu werben (z.B. bei der Projektevernissage, in Publikationen, auf der Homepage), wobei der/die Absolvent*in als Urheber zu nennen ist. Jegliche kommerzielle Verwertung/Nutzung bedarf einer weiteren Vereinbarung zwischen dem/der Studierenden/Absolvent*in und der FH St. Pölten.

Wösendorf,
07.09.2023



(Unterschrift Autor/Autorin)

Zusammenfassung

Die Vorhersage von Wasserständen und -durchflüssen ist wichtig, um sich beispielsweise zeitgerecht auf Überflutungen vorbereiten zu können oder den Schiffsverkehr auf größeren Flüssen planen zu können. Oftmals werden hierfür hydrologische Modelle eingesetzt. Diese Modelle müssen von Hydrolog:innen kalibriert werden, um eine zufriedenstellende Genauigkeit gewährleisten zu können. Diese Kalibrierung erfordert zum einen viel hydrologisches Wissen, zum anderen viel Wissen über das Einzugsgebiet des Gewässers.

Ein weiterer Ansatz zur Vorhersage von Wasserständen und -durchflüssen, welcher nicht so viel Domänenwissen voraussetzt, ist mittels Machine Learning (ML) Modellen. In der Literatur gibt es bereits zahlreiche Ansätze, die dieses Problem behandeln, sowohl mittels klassischer Regressionsmodelle als auch mithilfe von Deep Learning (DL) Modellen.

Aus diesem Setting resultiert folgende Forschungsfrage: Wie schneiden ausgewählte ML-Modelle im Vergleich mit einem hydrologischen Modell bei der Durchfluss-Vorhersage ab?

In dieser Arbeit wird zunächst eine Literaturrecherche über verschiedenste ML-Ansätze zur Vorhersage von Wasserständen und -durchflüssen durchgeführt. Anschließend wird eine Auswahl dieser ML-Ansätze (ARX und LSTM) implementiert und mit einem in der Praxis eingesetzten hydrologischen Modell verglichen. Das in dieser Arbeit als Benchmark verwendete hydrologische Modell wird aktuell vom hydrologischen Dienst des Landes Niederösterreich eingesetzt, um die Durchflussrate des Flusses Erlauf vorherzusagen. Zuerst wird die generelle Vorhersage-Genauigkeit über drei Prognosehorizonte evaluiert: 6 Stunden, 12 Stunden und 24 Stunden. Des Weiteren wird die Vorhersage-Performance bei Hochwasser-Ereignissen im Detail analysiert.

Die Auswertung der generellen Durchfluss-Prognosen zeigt, dass das hydrologische Modell akkuratere Vorhersagen liefert als die beiden getesteten ML-Modelle. Für den kürzesten Prognosehorizont von 6 Stunden konnten die ML-Modelle zwar noch gut mithalten, aber bei den Prognosehorizonten von 12 und insbesondere 24 Stunden waren deutliche Performance-Unterschiede zugunsten des hydrologischen Modells zu sehen. Vergleicht man die ML-Modelle untereinander, so liefert das simplere ARX-Modell die besseren Ergebnisse für die beiden kürzeren Vorhersagehorizonte, während das komplexere LSTM-Modell für den längeren Prognose-Horizont die besseren Resultate bringt.

Bei der Auswertung der Hochwasser-Ereignisse zeigte sich, dass das hydrologische Modell auch hier die deutlich besten Ergebnisse liefert, sowohl was die Vorhersage des Ausmaßes und des zeitlichen Eintretens des Spitzenwerts während eines Hochwassers betrifft als auch die generelle Modellierung des Hydrographen während eines Hochwasser-Ereignisses. Das LSTM-Modell schneidet hierbei nochmals merklich besser ab als das ARX-Modell.

Abstract

The prediction of water levels and flows is important, for instance, to be able to prepare for floods in good time or to plan shipping traffic on larger rivers. Hydrological models are often used for this purpose. These models must be calibrated by hydrologists to ensure satisfactory accuracy. This calibration requires a lot of hydrological knowledge on the one hand, and a lot of knowledge about the basin of the river on the other hand.

Another approach to predicting water levels and flows, which does not require so much domain knowledge, is by training Machine Learning (ML) models. Numerous approaches that address this problem have already been made and published in the literature, using both classical regression models and Deep Learning (DL) models.

This setting leads to the following research question: How do selected ML models compare to a hydrological model in water flow prediction in terms of accurate forecasts?

In this thesis, a literature review of different ML approaches for the prediction of water levels and flows is conducted. Then a selection of these ML approaches (ARX and LSTM) is implemented and compared with a hydrological model used in practice. The hydrological model used as a benchmark in this work is currently used by the hydrological service of Lower Austria to predict the flow rate of the river Erlauf. First, the general forecast accuracy is evaluated over three forecast horizons: 6 hours, 12 hours and 24 hours. Furthermore, the predictive performance during flood events is analysed in detail.

The evaluation of the general flow forecasts shows that the hydrological model delivers more accurate forecasts than the two tested ML models. For the shortest forecast horizon of 6 hours, the ML models were still able to keep up well, but for the forecast horizons of 12 and especially 24 hours, clear differences in performance in favour of the hydrological model could be seen. Comparing the ML models with each other, the simpler ARX model delivers the better results for the shorter two forecast horizons, while the more complex LSTM model produces the better results for the longer forecast horizon.

The evaluation of the flood events showed that the hydrological model also delivers the best results here, both in terms of predicting the extent and timing of the peak during a flood and the general modelling of the hydrograph during a flood event. The LSTM model again performs noticeably better than the ARX model.

Table of Contents

1 Introduction

Due to climate change, natural catastrophes such as floods will occur more and more frequently in the future [1]. On the other hand, low water levels can mean that shipping traffic will have to be restricted. The prediction of water levels and flow rates is therefore an important topic, for example to be able to warn people of floods or to plan shipping traffic in advance.

One approach to solve this prediction problem is the use of hydrologically based models. These models usually use information about various natural conditions, such as geographic location, soil properties or river basin characteristics to predict water flow. These models require a complex calibration to achieve satisfactory results, which also presupposes a lot of domain knowledge, that is often not available.

To circumvent the problem of this missing domain knowledge, Machine Learning (ML) has been proposed to forecast water flows and water levels. Many ML approaches have already been proposed, reaching from simple models like Linear Regression, Decision Tree or K-nearest Neighbour (KNN) [2] to complex Deep Learning (DL) models like attention-based Convolutional Long Short-Term Memory (Att-ConvLSTM) [3].

Most authors have compared their approaches against other, previously proposed ML approaches to show that the approach they came up with works better. One comparison that has not yet been done, however, is the comparison of ML approaches with a hydrologically based model which is already used in practice. This comparison is of great importance to be able to ensure that a ML model can really deliver satisfactory results. This master thesis attempts to carry out this comparison. Therefore, the research question of this work is: How do selected ML models, namely Autoregression with Exogeneous Inputs (ARX) and Long Short-Term Memory (LSTM) compare with a hydrological model in terms of accurate predictions in the field of water flow forecasting? With the help of this comparison, it can be analysed whether ML models are superior or inferior to expert systems, or whether it is a matter of judgement which models should be used.

The hydrological benchmark model used for comparison in this work is currently used by the hydrological service of Lower Austria to predict the water level and flow of the Erlauf river [4]. The measuring point “Niederndorf an der Erlauf” is used as the test station in this experiment.

The datasets used for training were also provided by the hydrological service of Lower Austria. Furthermore, theoretical input about hydrological concepts and the hydrological model was given by experts of the hydrological service of Lower Austria.

The master thesis deals with the prediction of water flow using different ML methods. The aim is to compare the two ML models mentioned above with a hydrological model. At first, those ML models will be implemented. After training the models, the evaluation against the hydrology-based model will be conducted. The evaluation will include the following seven metrics:

1. Mean Squared Error (MSE)
2. Mean Absolute Error (MAE)
3. Mean Absolute Percentage Error (MAPE)
4. R-Squared
5. Deterministic Coefficient (DC)
6. Error of Flood Peak Appearance (EPA)
7. Relative Error of the Flood Peak (REP)

Metrics 1-5 are general regression metrics, while metrics 6 and 7 measure the models' abilities to correctly model flood events.

The structure of this thesis is as follows: At first, related work is described in the *Systematic Literature Review* chapter. This chapter aims to give an overview of proposed ML models for water flow forecasting. Information about the river Erlauf, its basin and the datasets used in this thesis as well as the data preparation is given in the *Descriptions* chapter. Next, the ML models investigated, hydrologic models in general and the hydrological reference model as well as the model training processes are described in greater detail in the *Methodology* section. The evaluation strategy as well as the results are presented in the *Evaluation* section. The *Conclusion* section discusses the main findings and potential limits of this work. Furthermore, an outlook for potential further research in this area is given in this chapter.

2 Systematic Literature Survey

The following chapter summarizes previous approaches to predicting floods, water levels and flow rates of rivers in the form of a systematic literature survey. The literature was selected according to the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) framework [5]. This is to ensure that the results are made comprehensible and reproducible in the future.

The aim of this literature review is to summarize the current achievements and state of knowledge in the field.

2.1 Methods

The following databases were used as major sources to access relevant literature:

- IEEE Xplore
- ACM Digital Library
- arxiv.org

To obtain contextual literature, the following search query was used in the “Advanced Search” of the databases listed above:

("Flood Prediction" OR "Flood Detection" OR “Flood Forecasting“ OR “Predict Flooding" OR "Flood Mitigation") AND ("Machine Learning” OR "Artificial Intelligence" OR „Data Science“)

Please note that this is just the general notation of the query. Depending on the database, it must be adapted syntactically accordingly. Furthermore, only research papers published in 2010 or after were considered in this literature survey.

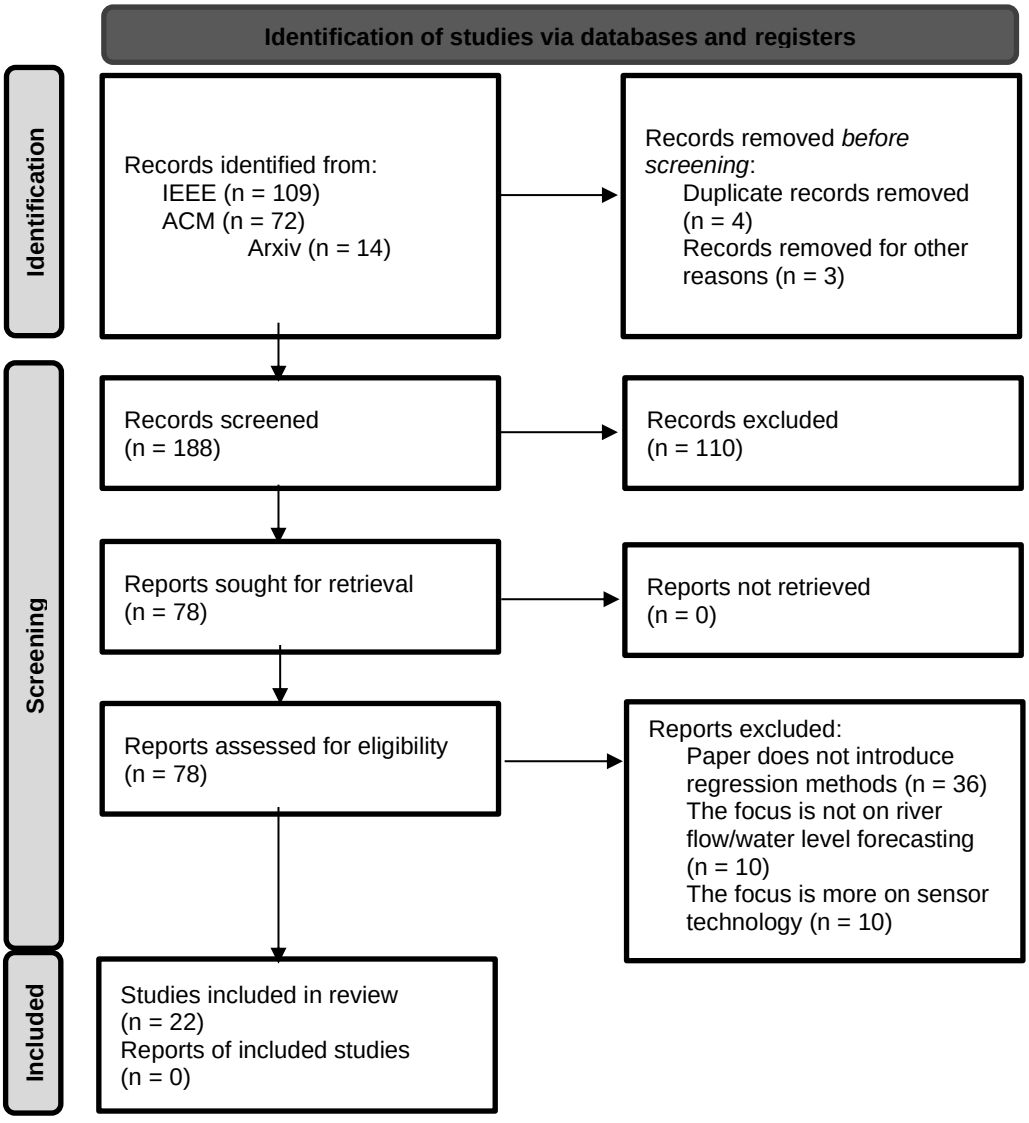
After obtaining the results, papers were screened and filtered by the following inclusion criteria:

- The paper needs to be written in English language.
- The word "flood" is understood in the context of natural disasters.
- ML is a core topic of the publication and not just a concept mentioned in passing.
- The goal of the paper is to forecast flooding/water level/flow rate rather than to nowcast it.

In the next step, the papers were checked based on the type of ML models in use. Since the model, which is used as a benchmark in this work, forecasts the water flow continuously rather than only distinguishing between flood and non-flood periods, only approaches that used regression models were considered.

2.2 Results

Various ML approaches to predict either flooding, river water level or river flow rate have already been made in the past. Based on the PRISMA workflow, the papers were screened and assessed for eligibility. The PRISMA flow chart [6] below shows how many papers were included or excluded.



In 2012, Chaowanawatee et al. [7] introduced a flood forecasting system that uses an artificial neural network (ANN) to predict the water level for Little Wabash River, USA. They used an architecture with multiple layers and made use of the Radial-Basis function as activation function. To optimize the model, the authors used Cuckoo Search based on Levy Flights instead of the backpropagation algorithm. Two types of Radial-Basis functions, namely Gaussian and Polyharmonic were compared. The evaluation of those two functions via rooted mean squared error (RMSE) revealed that the Polyharmonic function slightly outperformed the Gaussian function. In general, both models were not able to correctly detect abrupt fluctuations. Other than that, both models managed to predict the real water level appropriately. A risk of bias in this study is that the models were not evaluated against a baseline model, such as a hydrological model currently in use. Furthermore, it is not specified, which input features were exactly used to predict the water level.

Sanubari et al. [8] also used a Radial-Basis function-based ANN to predict the future water level for a river in Indonesia. They used data about precipitation and water levels from the past and present. The authors experimented with the number of hidden neurons and the learning rate. For evaluation, the mean squared error (MSE) was used. Their results showed that a neural network with 30 hidden neurons (i.e., Radial-Basis neurons) achieved the lowest error, while the optimal learning rate was 0.2. A potential risk of this study is, that the authors used a 50:50 split between training and testing data. A more commonly used ratio to split the data in training and testing is 80:20. The training data covered a time span of exactly one year. The first six months were used for training while the months July-December were used in testing, which means that the network was never trained with data from autumn or late summer.

Boukharouba et al. [9] proposed a flash flood forecasting system based on support vector regression for a river in the south of France. Instead of training one regression model, the authors trained multiple models on different data sets. The original data set was split based on similarity which was determined by hierarchical clustering. The goal of this approach was to obtain more specific models that should outperform a single global model. They used rainfall and previous water level data as input data. In total, five flash forecasting systems were trained for the prediction horizons of half an hour, one hour, two hours, three hours and four hours. The evaluation was done with the Nash coefficient and persistence and revealed that except for the half-hour prediction horizon, the multi-model approach was able to outperform the global model approach. One potential drawback of this study is that these models were specifically trained for flash floods and not for water level or flow in general.

Li et al. [10] used an approach based on support vector machines (SVM) and boosting learning algorithms for flood prediction at Huaihe River in China. As input data for the model, they used regional rainfall and runoff in upstream gauging stations. To reduce the dimensionality of the input data, Kernel Principal Component Analysis (KPCA) was applied. They compared an architecture consisting of multiple support vector machines based on the boosting learning to a single support vector machine. The evaluation of the models was done with MSE and deterministic coefficient (DC). The results revealed that the multi-SVM approach outperformed the single support vector machine especially in the flash flood cases, which the authors explain with the low occurrence of flash-flood-samples in the training dataset. They concluded that the boosting algorithm needs fewer training samples to output satisfying results. A potential risk of this study is, that there was again no baseline model involved in the evaluation.

A comparison of multiple ML algorithms for rainfall forecast and flood prediction in Wadi al Wala, Jordan, was conducted by Al-Fawa'Reh et al. [2]. Random Forest, Decision Tree, Linear Regression, K-nearest neighbour, and Support Vector Machines were compared to predict the water level. Input data from 13 rain gauge sensors was used, including humidity, rainfall, wind speed and temperature. For all metrics analysed (mean absolute error (MAE), MSE, R-squared and RMSE), the Random Forest and Decision Tree algorithms performed significantly better than the other three algorithms.

Liu et al. [11] combined stacked autoencoders and neural networks for the task of river water level prediction. A stacked autoencoder consists of one or more autoencoders followed by a regular feed-forward neural network. Like Boukharouba et al. [9], they trained a single model and a system consisting of multiple models that were trained on different datasets split based on clustering, in this case not hierarchical clustering but K-Means clustering. The authors used basin rainfall and upstream flows as inputs for their model. They compared their novel approach with a regular neural network, a neural network that uses radial basis functions, SVM and an extreme learning machine (ELM). Liu et al. showed that their approaches were able to outperform the other methods, according to the evaluation based on MSE and DC. Particularly in the periods with high water flow their models were able to reach better results. When comparing their single model with their pre-clustering multi-model system, the latter one achieved better results. Once again, these models were not tested against a hydrological model.

Ruslan et al. [12] compared two autoregressive models, namely Autoregressive with Exogenous Input (ARX) and Autoregressive Moving Average with Exogenous Input (ARMAX), which are two linear and parametric algorithms. As input, they used water level data from four other water level gauges. In evaluation, the models had to predict the correct water level in Pahang, Malaysia, with a seven-hour prediction horizon. The results showed that the ARMAX model achieved superior results compared to the ARX model. The evaluation however of this work is questionable, since the authors used approximately 1500 samples to train the models, approximately 2000 samples to validate and 4000 samples to test the models. Furthermore, these models were also not tested against a baseline model which makes comparison difficult.

In 2018, Wu et al. [13] combined hydrological and ML models and introduced the Long Short-Term Memory (LSTM) model to the domain of flood and water level forecasting. LSTMs, like recurrent neural networks (RNN) are neural networks that are designed to handle sequential data inputs. In addition, they used a so-called context-aware attention LSTM (CA-LSTM). Figure 2 illustrates the architecture of the CA-LSTM. Wu et al. claimed that this model should be better in determining which input factors are the most informative regarding the prediction. As input for the model, the factors evaporation, rainfall, flow rate and rain prognosis were taken. The two LSTM models were compared to a regular fully connected ANN and a support vector machine. Each model was trained for a prediction horizon of one to six hours. While the CA-LSTM was the best-performing algorithm for three, four, five and six hours in advance, the plain fully connected ANN actually outperformed its competitors for the short-term predictions of one and two hours. This suggests that the timespan to predict in advance impacts which algorithm should be chosen.

An attention-based LSTM-model for flood forecasting was introduced by Ding et al. [14]. The same four methods as mentioned in the work above were compared, however for a different location and river. As prediction horizons the authors specified three, six and nine hours. Interestingly, the results were similar to the results above: The fully connected model performed best for the three-hour prediction horizon while the attention-based LSTM model outperformed all other methods for six- and nine-hour prediction horizon. A potential risk here is that it is not specified, how the data pre-processing was done. Since LSTMs handle other inputs than regular neural networks, this should have been elaborated.

A comparison of RNNs and LSTMs for flood forecasting in Melbourne, Australia, was conducted by Gohar et al. [15]. The models were trained with three years of hourly river level, river flow and rainfall. The models were fed the last six hours of this information as inputs. Prediction horizons investigated reached from one hour to twelve hours. In the evaluation, MSE, MAE and RMSE were measured. For shorter prediction horizons, the models achieved similar performances but the longer the prediction horizon was set, the bigger the gap was between the two methods with the LSTM approach reaching better results.

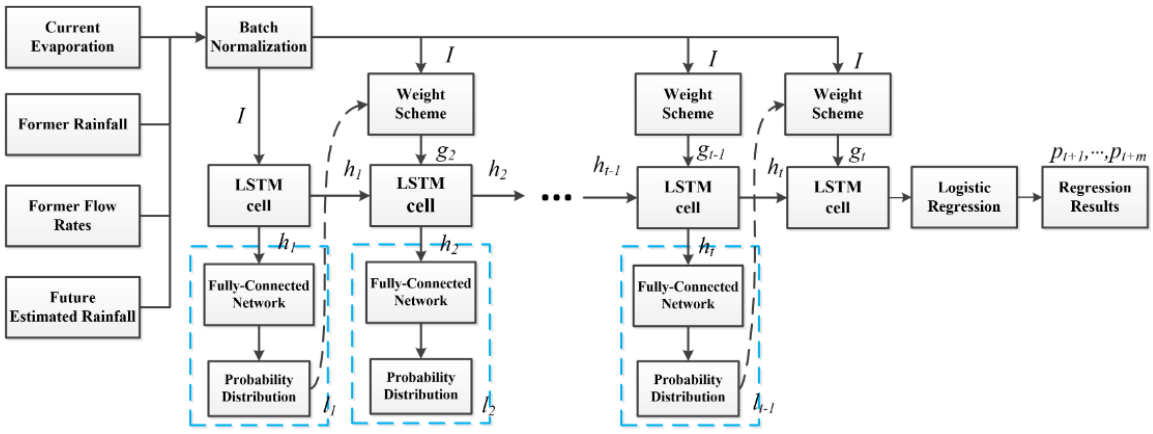


Figure 2: Architecture of the context-aware LSTM network from Wu et al.[13]

Liu et al. [16] proposed the combination of Empirical Mode Decomposition (EMD) and the encoder-decoder LSTM (En-De-LSTM) for water flow prediction for Yangtze River in China. EMD was used to pre-process the input of the model. The encoder-decoder LSTM consists of two stacked LSTMs: An encoding part, which encodes the input into a fixed-sized vector and a decoding part which decodes this vector into the original dimensional space. The model was compared to a vanilla LSTM model and a stacked LSTM. The evaluation based on R-squared and RMSE was able to outperform both of those approaches.

Another method, which is often used in as alternative to LSTMs to make predictions on time series data is the Gated Recurrent Unit (GRU). Selva Jeba et al. [17] combined GRU cells with RNN cells and Dense layers. This approach was compared to LSTM for heavy rainfall prediction based on wind speed and temperature in Kerala, India. The RNN-GRU approach outperformed the LSTM regarding RMSE and MAE. A potential risk of this study is however that the models were not used to predict water level but rainfall.

Miau et al. [18] combined the approach of GRU with the concept of convolutional neural networks (CNNs) to predict the water level of Danshui River basin, China. This model, which is called Conv-GRU, consists of convolutional layers, GRU layers and dense layers. Miau et al. trained a Conv-GRU and several other models (LSTM, CNN, ANN). For each approach, multiple architectures were trained. The best architectures of each model type were compared. The results revealed that Conv-GRU was able to outperform the other methods, which is interesting since both models on their own, CNN and GRU, were not able to predict the water level appropriately according to the authors. The evaluation of a vanilla GRU network cannot be found in the paper, which makes it difficult to verify this claim.

Wu et al. [19] proposed a flood prediction approach specifically for smaller rivers based on a hierarchical Bayesian network. The input factors used were evaporation, rainfall and information about the previous water level. An exact illustration of the model can be found in Figure 3. The evaluation was done for a prediction horizon of four hours. The model was compared to an ANN, a SVM, a Radial Basis Function network and an ELM. The evaluation was done on RMSE, DC, Error of Flood Peak Appearance (EPA) and Relative Error of Flood Peak (REP). Depending on the metric considered, once the RBF model and three times the Bayesian model presented was the best performing. This shows that the best-suiting evaluation metric must be chosen carefully.

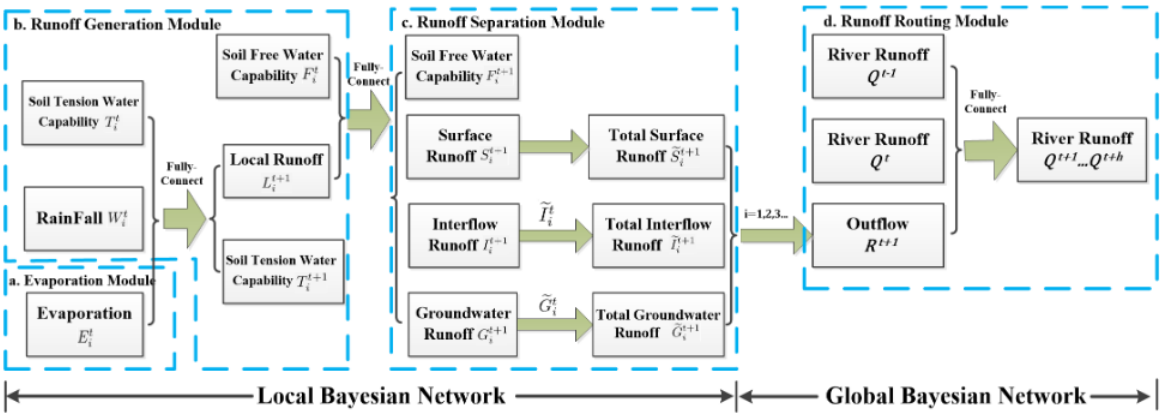


Figure 3: Architecture of the hierarchical Bayesian network from Wu et al.[19]

In 2019, Karyotis et al. [20] combined Deep Learning with the concept of fuzzy logic for flood prediction. Fuzzy logic is typically used to model uncertainty. The authors claim to have used input parameters that were a novelty to this date. They used inputs from sensors such as rain gauges, drain monitor sensors, meteorological data, information about geography and morphology and also input from an expert in hydrology. The Deep Learning model and the Fuzzy Logic model are connected sequentially. In the first stage, the Deep Learning model takes weather data, meteorological data and water level data as input and outputs the water volume forecast. More specifically, Karyotis et al. used a 1-dimensional convolutional neural network (CNN), which they state are an appropriate method for sequential input data. The authors used an input sequence of 200 measurements. The fuzzy logic model then used the output of the CNN and the remaining additional inputs to predict the probability of a flood occurring in 7 levels, reaching from extremely low to extremely high. Figure 4 gives a high-level overview of the models' architecture. A potential risk of this work is, that the second part, which means the fuzzy logic model, was not evaluated. Since this work aims to predict the water level of a river by using ML, which is what was done in the first part of this composed model, this approach is still worth mentioning.

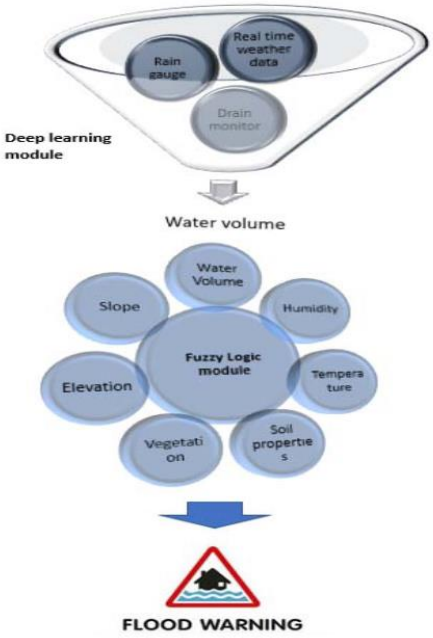


Figure 4: Deep-Fuzzy model from Karyotis et al. [20]

Wu et al. [21] discussed the fact that datasets about water levels are usually heavily imbalanced, which makes training an appropriate model usually more challenging. They addressed this problem by utilizing the Synthetic Minority Oversampling Technique (SMOTE). After balancing the dataset, they trained a Sparse Bayes Model based on AdaBoost. They compared a singular model and an ensemble model. The evaluation based on RMSE and DC revealed that the ensemble model outperformed the singular model in every single metric investigated. The number of models being included in the ensemble model is not included in their paper, which is a potential risk of this study.

Feng et al. [22] combined the concept of spatial and temporal learning to forecast water level. They used LSTMs to model the temporal aspect, Spatial Graph Convolutional Networks (S-GCN) to model the spatial aspect and a Hydrological Attention Module which consists of a 2-layer Multi-Layer-Perceptron (MLP) and Attention Scores. This module should help the model to focus on the most important input data. As inputs for the model, the authors used data about rainfall, evaporation, river flow and hydrological distance. The architecture of the model is illustrated in Figure 5. In their experiment, they tested the Spatial-Temporal Graph Convolutional Network (ST-GCN) against a K-nearest-neighbour (KNN), decision tree, LSTM, Independently Recurrent Neural Network (IndRNN) and against a S-GCN model. They evaluated their models on two rivers, namely the Chinese Tunxi and Changhua River. For both rivers, prediction horizons of one, three, six and nine hours were evaluated with RMSE and MAE. The results revealed that the ST-GCN performed best for the prediction horizons of six and nine hours in both rivers and also in the Tunxi river for three hours. For the Changhua River, the S-GCN, without the temporal aspect, slightly outperformed the ST-GCN for the three-hour prediction horizon. For the one-hour prediction horizon, in both cases the IndRNN model performed best.

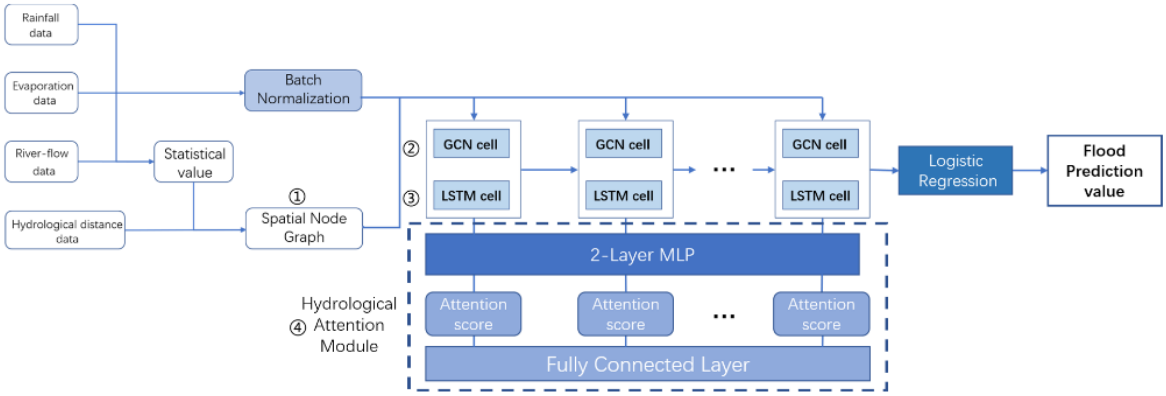


Figure 5: Illustration of the ST-GCN model (Feng et al.[22])

Nti et al. [23] compared four different ensemble ML techniques, namely LSTM, Extreme Gradient Boosting (XGBoost), Random Forest (RF) and Extra Trees (ET) to train flood forecasting models for the country of Ghana. ETs, like Random Forests, are an ensemble method and consist of multiple decision trees. The exact difference between Random Forests and Extra Trees lies in how the trees are exactly trained. The dataset used in this study consisted solely of rainfall data. The evaluation was conducted with the following metrics: MSE, RMSE, Mean Absolute Percentage Error (MAPE), R-squared and MAE. The evaluation revealed that XGboost and LSTM performed similarly well, while ET and RF performed significantly worse. One potential problem of this study is however that there was no detailed description of the models' architectures or the datasets.

A method to forecast the water level over a longer prediction horizon was proposed by Moishin et al. [24]. Combining the approaches of CNNs and LSTMs, they introduced ConvLSTM to forecast the water level on the Fiji Islands for one day, three days, seven days and fourteen days. This architecture was trained on mainly rainfall data and evaluated against a regular LSTM and a Support Vector Machine. For all forecast windows and all metrics investigated, the proposed method was able to outperform its competitor methods. A potential limitation of this work could be that the output variable was neither directly the water level nor the flow rate. Instead, a specific flood index was predicted, which makes it challenging to interpret the results.

Liu et al. [3] combined the ConvLSTM approach with the concept of attention and created Att-ConvLSTM. They proposed this technique to predict the water level of Arnoia River in Spain. The attributes used as input for the model were previous water level, previous water flow, temperature and rain. The authors compared their approach to an attention-based CNN-LSTM (Att-CNN-LSTM), an attention-based LSTM (Att-LSTM) and an attention-based bidirectional LSTM (Att-BiLSTM). The evaluation based on RMSE, MAE and R-squared showed that the authors' approach could outperform the other methods tested. While Att-CNN-LSTM and ATT-BiLSTM were only slightly worse, the ATT-LSTM was by far the worst-performing method.

Zhou et al. [25] introduced the concept of unsupervised flood prediction. This means that a model trained for one river can be reused for another river. The authors describe the approach the following way: At first, a regular, supervised prediction model is trained with labelled data. The prediction head is then detached from the encoding part. After this step, the so-called Adversarial Domain Adaption happens: The encoder trained by the source data is frozen. The goal of the encoder that is trained on the data from the target venue is to emulate the output of the source encoder. This is evaluated by a discriminator, which makes this concept similar to the concept of a Generate Adversarial Network (GAN). In the last step, the prediction head of the source data model is attached to the target encoder model. This model can now be used for inference. Figure 6 shows the architecture of the model. The approach was tested with datasets from two Chinese rivers, Tunxi and Changhua and evaluated by MSE and DC. The results showed that the unsupervised approach did not perform as good as a fully supervised approach but was able to outperform models that were only trained on parts of the data. This means that this approach can be useful in case there is not a big amount of target variables to train on but an already existing model trained on a different domain.

Zhu et al. [26] introduced an ML method for smaller rivers which normally lack sensor data compared to bigger rivers. The algorithms used to achieve this are called Spatial-Temporal Dynamic Time Warping (ST-DTW) and Multi-Feature algorithm. The models take soil water and rainfall as input. The data pre-processing of this approach involved rasterizing the data, in this case for example the rainfall was specified per square kilometre in the river basin. The authors compared their approaches to an LSTM, an SVM and an autoregressive integrated moving average (ARIMA) model for prediction horizons of one to ten hours. The analysis of the RMSE scores revealed, that ARIMA model performed the worst. The LSTM Model delivered the best results from one to seven hours prediction horizon. However, for the longer prediction horizons, the ST-DTW approach managed to outperform the LSTM model.

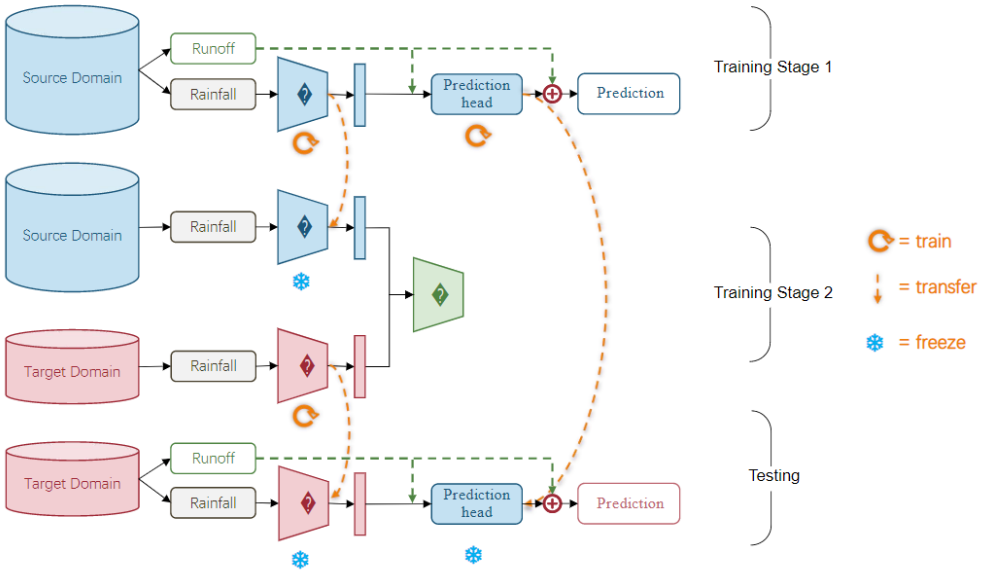


Figure 6: Illustration of the unsupervised adversarial model from Zhou et al. [25]

2.3 Summary and Discussion

This literature survey showed that there have been many different approaches to tackle the problem of water level and water flow forecasting with ML models, from classic models like linear regression to Deep Learning methods like numerous variations of LSTMs. However, none of those proposed models got compared to a hydrological approach during evaluation. For that reason, this is the research question of this work.

Depending on which location, prediction horizon or metric was used for the evaluation, different models ended up being the most suitable solution. This means that the prediction horizon of interest and the most relevant metrics must be determined carefully before starting this experiment.

Potential limitations of the research process used are:

- Not all databases available were used.
- The focus of research slightly changed while conducting this literature survey.

3 Descriptions

In this chapter, the characteristics of the river Erlauf and its river basin, as well as the datasets and the data preparation are described. This is done to make it easier for further research works in this area to assess whether a river is comparable to the river Erlauf. Furthermore, the datasets used for model training are described. An overview over each dataset is given and the most important data pre-processing steps are explained.

3.1 River and River Basin Description

The river Erlauf is located in Lower Austria and is a tributary river of the river Danube. It originates in the mountainous South of Lower Austria on the border with Styria and is approximately 67 km long [27]. Its river basin covers an area of about 630 km^2. Based on yearly water reports from 2004 to 2013, the average yearly precipitation in the river basin is approximately 1400 liters per square meter (l/m^2). 850 l/m^2 contribute to runoff while 550 l/m^2 contribute to evaporation. Other factors such as water stored in form of snow or soil humidity typically vary throughout the year but do not change when looking at periods from September to August (which are also called “hydrological years”) to exclude high water storage due to snow or high soil moisture as done in [28].

While the upper course of the river is surrounded by mountainous, forested terrain, the middle course of the river flows mainly through grassland and the lower course largely through farmland. Furthermore, two hydroelectric power plants and three associated reservoirs are located in the upper course of the river, which can influence the current flow and runoff in this area. Due to this and the overall lower water flow, it is generally more difficult to make reliable forecasts in the upper river course. Therefore, the gauging station “Niederndorf” is the point of interest in this work. It is located in the lower course of the river and the last gauging station before the river flows into the river Danube [28]. The average water flow at this station is 14.6 cubic meters per second (m^3/s). The yearly flood value, which will be of interest in the evaluation, is 170 m^3/s [29].

In this work, a flood event is defined as follows: First, all time periods were searched in which the annual flood value (HQ1) of Niederndorf, which is 170 m^3/s, was exceeded in the observed dataset. Each period was then extended forward and backward by 12 hours to be able to assess the models’ capabilities to correctly detect the rise and fall of the hydrograph. If two time periods overlapped, they were merged into one time period, which effectively means that if there were 24 hours or less between two flood periods, they were combined into one flood event. This will be relevant later in the *Evaluation* section of this work.

3.2 Dataset Description

Various data sources were used to train models to predict the water flow for the river Erlauf at Niederndorf and evaluate them: Precipitation and temperature from the river basin and water flow rates from Niederndorf as well as a benchmark dataset for the hydrological reference model. The data was provided by the hydrological service of Lower Austria.

3.2.1 Precipitation and Temperature Data

Raster data about the river basin's precipitation and temperature was used. The data originates from GeoSphere Austria [30]. The raster data for temperature and precipitation was received for the whole of Lower Austria with raster sizes of 1x1km, which resulted in a matrix of 231x200 fields. Since only the river basin of the Erlauf river was needed, a reference raster was used to extract the required raster cells. Figures 7 and 8 show an example of a rain grid data point and the reference rasters. Both figures were created using the Python package matplotlib [31]. The temperature was measured in degrees Celsius and the rainfall was measured in l/m^2. The data ranges from 01/01/2003 to 31/12/2016 with one observation every 15 minutes.

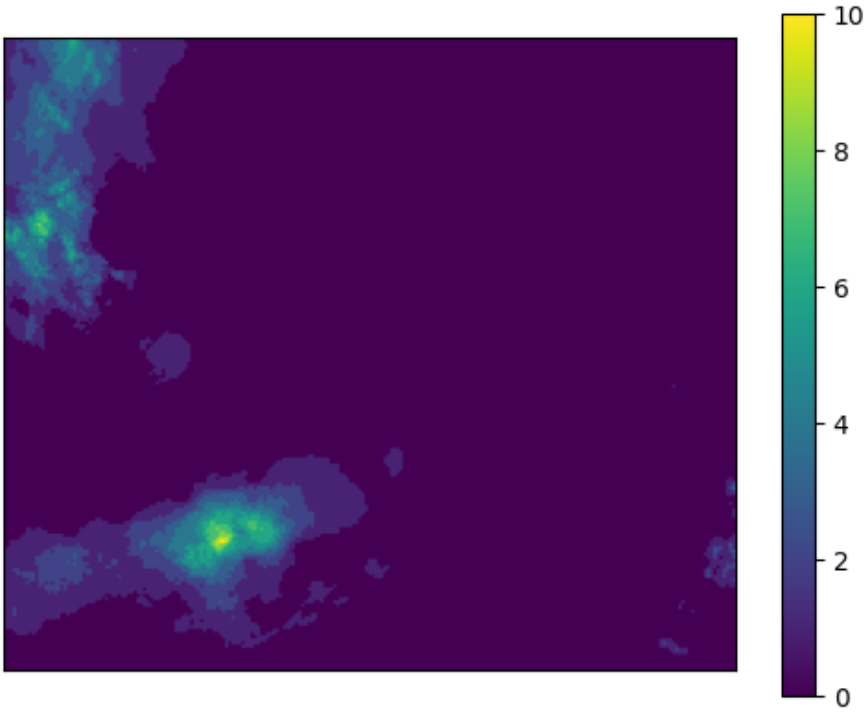


Figure 7: Example of a rain grid data point

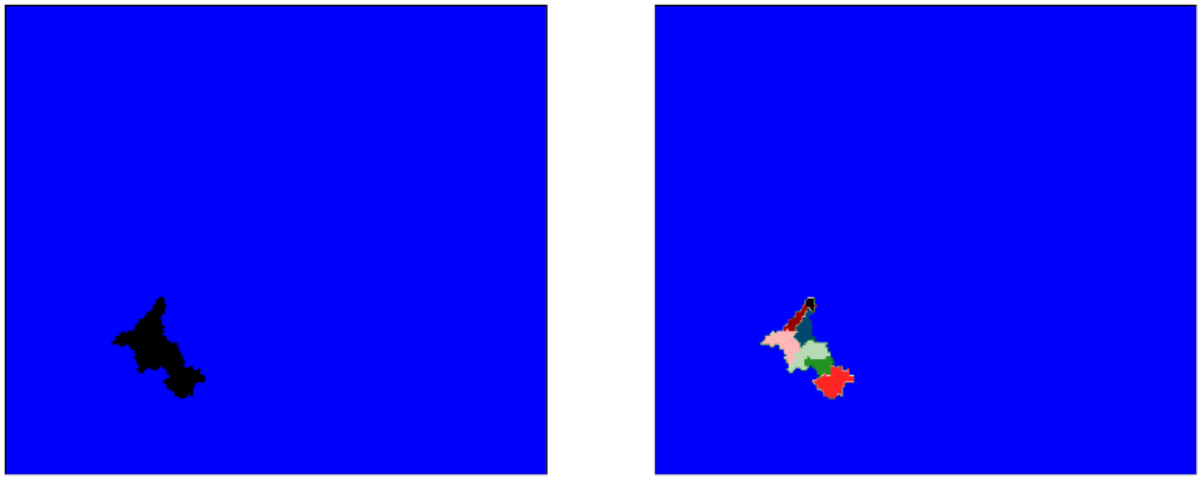


Figure 8: Reference Raster for the total Erlauf river basin in the Lower Austria Grid (left) and the subbasins (right)

3.2.2 Benchmark Data

The benchmark dataset was used to assess the predictive performance of the hydrological benchmark model. It essentially consists of three columns: timestamp, simulated water flow and observed water flow. The water flow was measured in m^3/s. The output correction, which is described in chapter 4.3, is not yet included in this data set and was manually added with Python during data preparation. The data ranges from 01/09/2003 to 31/12/2022 with one observation per hour. The source of this dataset is Afry [32] [33].

3.3 Data Preparation

The data preparation was done with Python 3.9 [34]. The following packages were used: pandas [35] for basic tabular data pre-processing, rasterio [36] for reading raster data, numpy [37] for basic array operations and pickle, which is a built-in Python package, for storing intermediate and final datasets as feather files. Since the data sets have different start and end times, the first step was to align the start and end dates of the respective data sets. Because observations across all datasets were available from 01/09/2003 to 31/12/2016 those date ranges were used in this work.

The raster data for temperature and precipitation was available for the whole federal state of Lower Austria. Since only the Erlauf river basin from the river origin to Niederndorf was the region of interest, the values had to be extracted. This was done via a reference raster, which was also received from the hydrological service of Lower Austria. To further lower the dimensionality of the data, the data was aggregated on river subbasin level. For both precipitation and temperature, the mean value was used. This was also done via reference grids for the 7 subbasins of interest. This brought down the number of data points from 46200 to 7 data points per timestamp (7 for temperature and 7 for precipitation). Instead of using the precipitation and temperature values from a specific point in time, the average of the last 48 hours per time step was used.

To handle the different granularities between the datasets (one observation per hour in the benchmark dataset and four observations per hour in the raster data), the raster datasets were aggregated to the level of one observation per hour. This was done by taking the respective average values for temperature, precipitation and water flow.

In order to be able to compare the ML models with the benchmark model in real-time-like setting, the output correction was applied using Python. The mechanism of the output correction is described in chapter 4.3.

After the preparation, the datasets were merged on the “timestamp” column. The data was split into train and test set on the date of 31/12/2013. Everything up to this date was used for training and validation and the remaining three years of the dataset were used for testing, which means the split ratio between training/validation and testing data was approximately 80:20. Within the training/validation time series, eleven flood events are present, while the test time series contains five flood events. Table 1 shows an extract from the final data set.

Timestamp (UTC)	Flow (t-1)	Precipitation Kienberg	Temperature Kienberg	Precipitation Scheibbs	Temperature Scheibbs
2006-08-07 22:00:00	239.24	1.072304	10.901080	0.864512	11.868040
2006-08-07 23:00:00	245.90	1.099219	10.948215	0.887277	11.902306
2006-08-08 00:00:00	258.34	1.100651	10.991996	0.888210	11.933086
2006-08-08 01:00:00	264.83	1.097618	11.037875	0.881279	11.964835
2006-08-08 02:00:00	267.35	1.085049	11.077734	0.869481	11.995770

Table 1: Extract of the final data set used for model fitting

4 Methodology

The following chapter describes the investigated ML methods as well as the hydrological model which is currently in use by the hydrological service of Lower Austria for predicting water flow rates at the Erlauf river. To cover a broad horizon of ML models, one model from the traditional statistical model family and one model from the more complex, Deep Learning family was chosen for this work. The ARX model was chosen because it is the simplest time series related ML model. The LSTM model was chosen because it was used in various works found during the systematic literature survey.

During the literature review, some approaches which use both hydrological and ML aspects were found. This work aims to focus on pure ML models because the goal is to investigate whether it is possible to train an ML model with satisfactory results without any hydrological domain knowledge or too much knowledge over the characteristics of the river basin.

4.1 Autoregression with Exogeneous Input (ARX)

The Autoregression with Exogeneous Input (ARX) model is one of the simplest time series models. It aims to predict the next value of a time series by finding the optimal combination of the past values of the time series and exogeneous features. As with a conventional linear regression, the optimal linear combination is determined in the fitting process by minimising the error between the predicted and actual values (for example, using Ordinary Least Squared). Basically, the ARX model is a linear regression but by using information about the past values of the time series rather than only using exogeneous features, it is more suitable for time series problems than a conventional linear regression. An example of an ARX model using three exogeneous variables and two previous time steps can be found below:

$$\widehat{y(t)} = \beta_0 + \beta_1 * x_1(t) + \beta_2 * x_2(t) + \beta_3 * x_3(t) + \alpha_1 * y(t-1) + \alpha_2 * y(t-2)$$

β_0 represents the model's intercept, β_1- β_3 represent the coefficients for the exogeneous variables (x_1- x_3) and α_1- α_2 stand for the coefficients for the lagged target variables ($y(t-1)$ and $y(t-2)$).

The ARX model was used to predict the water level of the Pahang River in Malaysia by Ruslan et al. [12]. It was compared with a related linear method, the ARMAX (Autoregressive Moving Average with Exogeneous Input). In the work of Ruslan et al. the ARX model achieved the worse results. Nevertheless, in this work the ARX model is analysed in order to represent as wide a range of ML models as possible, from very simple to more complex. In this work, the ARX model represents the simplest ML approach to predicting water flow.

4.2 Long Short-Term Memory (LSTM)

Recurrent Neural Networks (RNNs) were the first neural networks specifically designed to solve time series forecasting problems. In terms of structure, an RNN shares a lot of properties with a regular feed-forward neural network. The main difference is that in an RNN there are connections between the hidden layers with respect to time. This can be imagined as follows: The data point from time t-2 is passed to the input layer and then to the hidden layer. In parallel, the data point from time t-1 is transferred to the input and then to the hidden layer. There is a connection between the hidden layer from time t-2 and the hidden layer from time t-1 in order to be able to capture temporal relationships between the data points. The weights of the connections are learned through the so-called backpropagation through time (BPTT) algorithm [38]. Figure 9 illustrates the architecture of an RNN, the compact notation on the left and the unrolled version on the right. The green circles correspond to the input layer, the blue rectangles represent the hidden layer with its temporal connections. The orange circles at the top depict the output [15].

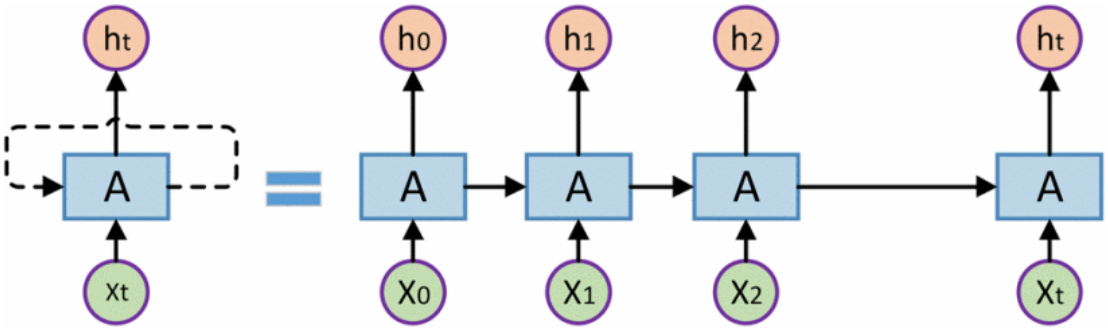


Figure 9: Illustration of a simple RNN[15]

However, a disadvantage of RNNs and the BPTT algorithm is that as the number of time steps (i.e., number of inputs) increases, the gradients either explode (for weights greater than 1) or vanish (for weights less than 1). This behaviour, which is also called the “vanishing/exploding gradient problem”, means that RNNs have difficulties learning long-term dependencies or features [39].

Hochreiter and Schmidhuber [40] invented the Long Short-Term Memory (LSTM) model to tackle this problem. In contrast to RNNs, LSTMs have two different paths were information flows, one for short-term and one for long-term information. One so-called LSTM cell is illustrated in Figure 10. The input vector xt updates both the short-term and long-term memory. The change of the long-term memory is depicted in the top path from C_{t-1} to C_t. The change of the short-term memory is illustrated in the bottom path between h_{t-1} and h_t [15].

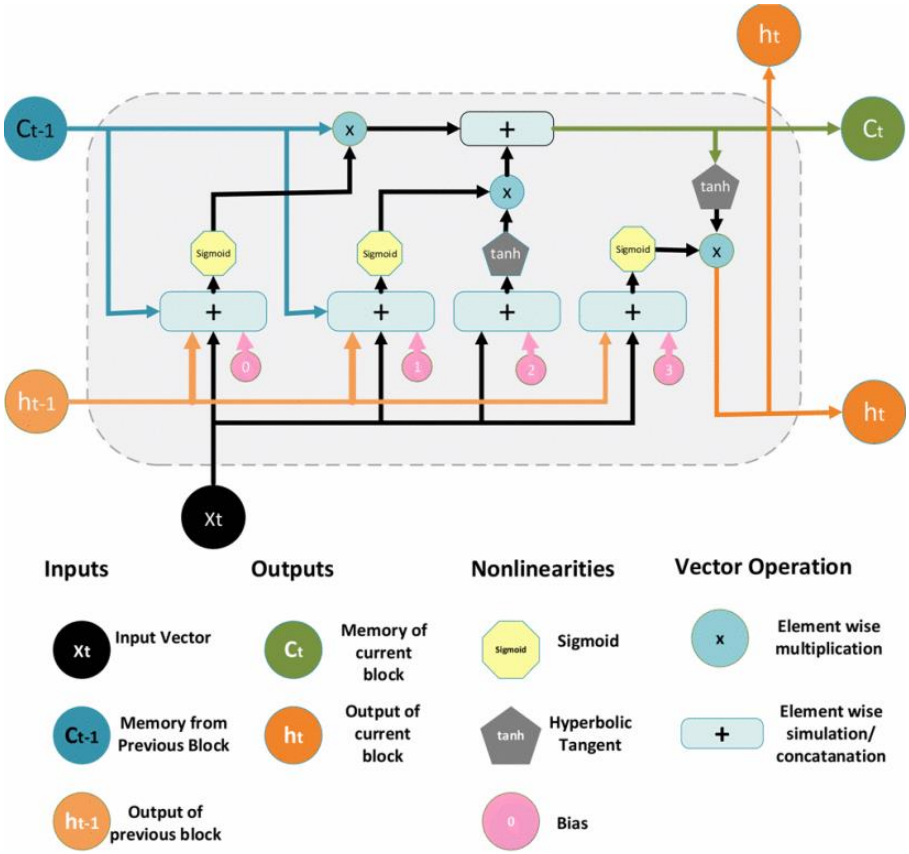


Figure 10: Illustration of an LSTM cell [15]

4.3 COSERO Model

Hydrological models can have many different objectives. In addition to predicting water flow, these include strengthening the understanding of a hydrological system or serving as a basis for decision-making for future water management projects. Based on their properties, hydrological models can be divided into subcategories according to Pechlivanidis et al. [41].

Firstly, hydrological models can be classified based on their structure. The first group of hydrological models are metric models or sometimes also referred to as empirical models. Metric models are typically based on observations from the past and try to model water flow based on empirical data [41]. Metric models do not care about underlying hydrological processes, they simply model input (in most cases precipitation data) to output (typically runoff or water flow), which means they require a large amount of observed data. ML models used for water flow forecasting fall under this category [42].

The second group are conceptual models or sometimes also called parametric models. Conceptual models usually try to simulate the main processes of the whole hydrological cycle by taking for example precipitation as input and computing soil moisture, evaporation, potentially snow cover and runoff or water flow in the end. Model parameters are not only obtained from observations from the past but also from manual fitting (in the context of hydrology often called “calibration“) [42].

The last group are physically based models. Like conceptual models, physically based models try to model the complete hydrological cycle. However, the architecture is typically more complicated. Here, governing equations of motion are used to estimate all components of the water cycle appropriately. Many models do not fall into exactly one of those categories and are therefore often referred to as hybrid models [41].

Another way hydrological models can be divided is based on their regional granularity. Typically, a distinction is made between lumped models and distributed models. Lumped models assume that the entire river basin is one large, homogeneous element. There is no information about different altitudes or other properties such as different geometric features. In distributed models, the river basin is divided into smaller, uniformly sized units (e.g., grid squares). For each unit, specific features such as runoff is calculated and aggregated together at the end. In contrast to lumped models, specific regional information can be considered. On the other hand, a lot more information is needed for such a type of model. A third category, which sits in between lumped and distributed is called “semi-distributed”. Here, the river basin is also divided into smaller elements, but not by a grid, but rather by several lumped models, where each lumped model represents for example a specific subbasin of a river. Thus, different features of different regions can still be leveraged, but the complexity of such a model is not as high as of a distributed model [41].

Based on the regularity of input, a hydrological model can either be continuous or event based. While a continuous model receives new inputs in each specified time step (e.g., hourly or daily), an event-based model only takes into account specific events such as storms [41].

The hydrological model used as a benchmark in this work is currently used by the hydrological service of Lower Austria and is called “COSERO” model. “COSERO” is an abbreviation for “Continuous Semi-Distributed Runoff”. As the name already suggests, this model belongs to the family of the continuous and semi-distributed hydrological models. Furthermore, it is stated to be a conceptional model [28]. The COSERO model is also used for various other rivers. Depending on where the river is located, the model’s overall architecture can vary. For rivers located in Europe, precipitation, temperature and potential evapotranspiration are taken as input. Based on whether the temperature is positive or negative, the model assigns the precipitation at a specific timestamp to liquid water or to snow cover. If the temperature becomes positive, snowmelt is computed. Based on potential evapotranspiration and interception (i.e., the amount of precipitation, that is absorbed by plants) the actual evapotranspiration is calculated. The remaining rainwater is divided into infiltrating and draining water. The latter is the main source for calculating the water flow. In addition to this precipitation-runoff model, the COSERO model uses a so-called “Water Allocation Model” (WAM). This model enables the correct routing of the hydrograph downstream from one gauging station to the next [43]. The COSERO model thus does not have a single target variable but attempts to model the entire water cycle in a river basin. Therefore, it does not have an autoregressive part, such as an ARX model. However, to obtain the best possible predictions in operational systems, the so-called output correction is used. This means that if the simulated flow value of the COSERO model is off the observed flow value by a specific amount at the current time step t, the predicted hydrograph is shifted. Based on the runoff class, the hydrograph is either shifted in parallel by the error value, or so that the correction for time step t+1 is the error value and then decreases more and more per time step into the future.

In order to be able to predict the flow value for a horizon of, for example, 6, 12 or 24 hours, the according precipitation and temperature forecasts are needed. Since the COSERO model is semi-distributed, the Erlauf river basin was divided into smaller areas. This was done by firstly dividing the basin into sub-basins and then dividing the sub-basins based on elevation and other natural characteristics as geology or soil morphology into so-called hydrological response units (HRUs). The parameter calibration was done partly manually and partially automated. Since there are multiple gauging stations along the river, it had to be decided, which stations are more important to model correctly. The gauging station of Niederndorf, which is the subject of interest in this work, was one of them [28].

4.4 ML Model Fitting Processes

All models were trained using Python 3.9 using the same features as the COSERO model: precipitation, temperature and past flow values of the station. Before training, all input features were standardized, meaning every feature had a mean of 0 and a standard deviation of 1. This was done using the scikit-learn [44] package from Python.

For each ML model, different hyperparameters had to be tuned. Therefore, the training dataset was further split into training and validation. The years 2003-2012 were used for training and the year 2013 was used for testing the different hyperparameter combinations. Once the optimal hyperparameter combination was found, the training and validation datasets were merged, and the final model was trained.

4.4.1 ARX Model Training

The ARX model was trained using the Python package skforecast [45], which also supports doing grid search for longer prediction horizons. Since the ARX model essentially is a linear regression, there are not many hyperparameters to tune besides the number of prior observations and if an intercept is used or not. Table 2 shows the hyperparameters, the values tested and the optimal values which were used later to fit the final model.

ARX Hyperparameters		
Hyperparameter	**Values tested**	**Optimal**
No. of Prior Observations	1-100	45
Fit an intercept?	True, False	True

Table 2: Tested ARX hyperparameters

4.4.2 LSTM Model Training

The LSTM model was implemented via the Python package keras [46]. Since LSTM models take significantly longer to train than linear models like ARX, the optimal hyperparameter search was done manually rather than with grid search to save time. Furthermore, unlike skforecast, keras does not support model fitting for longer prediction horizons in a rolling forecast style, which is why during training, a forecast horizon of only one hour could be used. The LSTM models were trained for 80 epochs, but an early-stopping mechanism was integrated, which ensured that the training stops if no further improvement is achieved over 10 epochs. Furthermore, the models' weights, which resulted in the optimal validation loss were used rather than the lowest training loss. The loss monitored during training was MSE. Table 3 shows all hyperparameters tested as well as the optimal values.

LSTM Hyperparameters		
Hyperparameter	**Values tested**	**Optimal**
Learning Rate	0.01, 0.02	0.02
Dropout Ratio	0.2, 0.3	0.2
No. of LSTM Layers	1, 2, 3, 4	3
No. of LSTM Cells per Layer	5, 10, 20, 32, 64, 100, 200	100
No. of Feed Forward Layers	1, 2, 3	2
Batch Size	16, 32	16
No. of Prior Observations	10, 25, 50, 75, 100, 200	100

Table 3: Tested LSTM hyperparameters

The architecture of the trained LSTM models can be described the following way: The input layer is followed by the LSTM layers. Between two LSTM layers there is a Dropout layer. The block of LSTM and Dropout layers is followed by Dense layers. The last Dense layer has one single neuron and is thus also the output layer.

One problem that occurred during the LSTM training process is that many models were prone for small errors at t+1 adding up drastically and resulting in enormous errors at t+24. This is probably because in keras, it is not possible to train a rolling forecast model that uses its own predictions as inputs for the next time step.

5 Evaluation

The following chapter describes the evaluation approach and the metrics that were used in the process. Furthermore, the results of the evaluation are presented.

The models were assessed by regular regression metrics on the one hand and by flood specific metrics on the other hand. Each model made a prediction for each time step (i.e., every hour) starting from 01/01/2014 at 00:00 until 31/12/2016 at 23:00. Each prediction had a horizon of 24 hours. The predictions were obtained in the rolling window forecast style: The model predicted the value for t+1 and used this value as input for t+2. The predicted and actual values were compared via five regression metrics: MSE, MAE, MAPE, R-squared and DC. Three different prediction horizons were evaluated: 6 hours, 12 hours and 24 hours. Furthermore, the flood events of the test dataset were also numerically evaluated using two metrics (REP and EPA) and visually evaluated.

5.1 Regular Regression Metrics

The metrics used for evaluation can be divided into two subcategories: Traditional regression metrics and flood specific metrics.

Traditional regression metrics used were MSE, MAE, MAPE and R-squared. MSE measures the averaged squared distance between the predicted and the measured values. MAE measures the average absolute distance between the predicted and the actual values. MAPE measures the percentage deviation between predicted and the observed values. R-squared is the square of the Pearson correlation coefficient. It measures the amount of the variance of the observed values that is captured by the predicted values. Additionally, the deterministic coefficient (DC) was implemented because it was found in various papers during the literature review as for example in Zhou et al. [25] and Wu et al [19]. According to Zhou et al., DC is a good indicator for how well a trained regressor fits the data. The DC is a coefficient between negative infinity and 1, the higher the better [25]. The formula of DC can be found below.

$$DC = 1 - \frac{\sum_{i=1}^{t}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{t}(y_i - \bar{y})^2}$$

The letter t represents the number of observations (i.e., timesteps) present in the dataset, $\hat{y}_i$ stands for the predicted water flow value at timestamp i. yi is the actual water flow value at timestamp i and $\bar{y}$ represents the average water flow throughout the whole time series [25]. Note that this definition is identical to the often-used Nash-Sutcliffe Efficiency (NSE) [24].

5.2 Flood Specific Metrics

Floods are the most tragical events that can happen related to water flows and water levels. Therefore, two metrics specifically for floods were also considered. The first metric is called “Relative Error of the Flood Peak” (REP). It measures the relative distance between the predicted peak value and the actual peak value. The other metric is called “Error of Flood Peak Appearance” (EPA). EPA also measures correct prediction of the flood peak, but rather than looking at the value of the flood peak, it measures the error of the appearance time. Both metrics were also used for evaluation by Wu et al [19]. The formulas can be found below.

$$REP = \frac{1}{n}\sum_{j=1}^{n}\frac{|pj - rj|}{rj} \times 100\%$$

$$EPA = \frac{1}{n}\sum_{j=1}^{n}|dj - sj|$$

In both formulas, n represents the number of flood events in the dataset. In the REP formula, pj stands for the predicted highest water flow value during one flood event and rj is the real highest water flow value during the same flood. In the EPA formula, dj represents the timestamp of the predicted flood peak and sj stands for the timestamp of the observed flood peak [19]. In this work, EPA will be measured in hours, since it is the smallest unit of time available.

5.3 Results

The evaluation was done by the metrics described in Section 5.1 and Section 5.2. For MSE, MAE and MAPE the implementation of the Python package scikit-learn [44] was used. For the R-squared metric, the square of the Pearson correlation coefficient function of the Python package scipy [47] was used. The metrics DC, EPA and REP were implemented from the ground up.

In addition to the numerical evaluation, the flood periods of the predicted and observed hydrographs are visualized below to gain more knowledge about the models' capability to correctly model the flood events in terms of duration and intensity.

For more context of the results, a baseline model which always predicts the mean value of the water flow time series (training set) was also evaluated. To be able to evaluate against the ground truth and the benchmark model, the previously conducted standardization was reverted before evaluation.

On some occasions, the models (also the hydrological model after applying the output correction) predicted negative flow values. Since negative flood values are physically not possible, all negative values were set to zero.

5.3.1 6-hour Prediction Horizon

For the 6-hour prediction horizon evaluation, the first 6 hours of each 24-hour-prediction were evaluated against the observed values of the next 6 hours of the water flow time series. The baseline model, which always predicted the mean value of the training time series significantly performed the worst, which means that the three other models managed to grasp the basic features of the hydrologic measures. The R-Squared of 0 and the DC of negative infinity indicate that the mean-value predictor does not fit the data well, especially regarding the variance, which was expected. The COSERO model achieved the best scores across all five metrics. However, the ARX model came fairly close in every metric, especially MAE, R-squared and DC. R-squared and DC indicate that the ARX model managed to correctly reproduce most of the variance of the data, scoring 0.95 in both metrics. The LSTM model also got over 0.9 in both scores, which also indicates a good performance, but interestingly got worse results than the ARX model across all five categories, despite being a more complex model with more parameters. Especially in terms of MAE, the significantly worse performance of the LSTM model is interesting considering the ARX model was almost on the same performance level as the COSERO model. Table 4 shows the performance scores of all models investigated for the 6-hour prediction horizon.

Model	MSE ↓	MAE ↓	MAPE ↓	R-squared↑	DC ↑
Baseline	323.27	8.75	80.70%	0.00	-inf
COSERO (Hydrological)	8.91	0.90	4.97%	0.97	0.97
ARX	15.43	0.99	6.01%	0.95	0.95
LSTM	27.03	1.53	9.87%	0.92	0.91

Table 4: Performance scores for a 6-hour prediction horizon

5.3.2 12-hour Prediction Horizon

As with the 6-hour prediction horizon, for the 12-hour prediction horizon evaluation, the first 12 hours of each 24-hour-prediction were evaluated against the observed values of the next 12 hours of the water flow time series. The performance scores of the baseline model barely changed, which was expected. The models investigated still all managed to significantly outperform the baseline model. The COSERO model did not lose a lot of performance despite the more difficult task of predicting a 12-hour window, especially regarding R-squared and DC. The ARX and LSTM models both lost significantly more performance across all five metrics, which means that the COSERO model was the best performing model in this prediction horizon by a larger margin compared to the 6-hour horizon. Interestingly, the ARX model still just outperformed the LSTM model. On the one hand, looking at the MSE, the performance difference was small. On the other hand, looking at the MAE and MAPE, the difference seems a little bigger, which suggests that the LSTM model struggles especially predicting lower values accurately. Table 5 shows the performance scores of all models investigated for the 12-hour prediction horizon.

Model	MSE ↓	MAE ↓	MAPE ↓	R-squared↑	DC ↑
Baseline	323.27	8.75	80.69%	0.00	-inf
COSERO (Hydrological)	17.50	1.31	7.04%	0.95	0.95
ARX	45.97	1.79	11.08%	0.86	0.84
LSTM	50.48	2.27	14.71%	0.84	0.81

Table 5: Performance scores for a 12-hour prediction horizon

5.3.3 24-hour Prediction Horizon

For the 24-hour prediction horizon evaluation, all values of each 24-hour-prediction were used. As expected, the performance metrics of the mean-predictor baseline model barely changed. All three models were still able to significantly outperform this baseline, which is a sign of good model quality. The COSERO model still only lost a slight amount of performance despite the prediction horizon doubling in length again, with R-squared and DC still being above 0.9, MAPE still being below 10% and MSE/MAE only increasing slightly compared to ARX and LSTM, meaning that it also got the best results in this category. The ARX model lost the most performance compared to the 12-hour prediction horizon. The LSTM model managed to outperform the ARX model in three performance metrics: MSE, R-squared and DC. The latter two indicate, that the LSTM model was able to fit the data better for longer prediction horizons compared to the ARX model, which is something that was also discussed in many works in the literature survey: Simpler models can work better for shorter horizons while more complex models have the advantage in longer prediction horizons. However, it is worth noticing that the ARX model still had a lower MAE and MAPE, which indicates a better performance for lower flow values. Table 6 shows the performance scores of all models investigated for the 24-hour prediction horizon.

Model	**MSE ↓**	**MAE ↓**	**MAPE ↓**	**R-squared↑**	**DC ↑**
Baseline	323.26	8.75	80.67%	0.00	-inf
COSERO (Hydrological)	26.96	1.76	9.33%	0.93	0.93
ARX	95.44	2.96	19.61%	0.70	0.62
LSTM	86.00	3.31	22.79%	0.73	0.71

Table 6: Performance scores for a 24-hour prediction horizon

5.3.4 Flood-specific Analysis

In total, five flood events were identified within the test dataset. A flood was defined as a time period, where the HQ1 value, which indicates the annual flood value (170 m^3/s), was exceeded plus 12 hours before and after that time period. Each models' prediction at this timestamp was used as the starting point of the prediction. The whole flood period had to be predicted in one go, which resulted in prediction horizons longer than 24 hours. Table 7 shows the time, duration and peak value for each flood event present in the test dataset.

Flood No.	Start-End	Duration	Peak Value
1	15/05/2014 16:00 – 18/05/2014 02:00	58 hours	399.39 m^3/s
2	27/05/2014 23:00 – 29/05/2014 07:00	32 hours	257.61 m^3/s
3	23/10/2014 04:00 – 24/10/2014 16:00	36 hours	243.30 m^3/s
4	10/01/2015 21:00 – 11/01/2015 23:00	26 hours	178.05 m^3/s
5	01/02/2016 02:00 – 02/02/2016 03:00	25 hours	171.33 m^3/s

Table 7: Properties of all flood events present in the test dataset

The metric EPA measures by how many hours the predicted flood peak was missed per flood on average. The COSERO model missed the flood peak on average by less than half an hour. Both the ARX and the LSTM model missed the flood peak by more than half a day. The metric REP measures the relative error between the predicted and the measured flood peak value. The COSERO model also performed the best in this category by a large margin. In this case, the LSTM model also managed to outperform the ARX model significantly. Table 8 shows the performance metrics regarding flood peak detection for each model investigated.

Model	EPA (hours)	REP
COSERO (Hydrological)	0.4	11.60%
ARX	19.0	73.30%
LSTM	14.6	44.11%

Table 8: Performance metrics regarding flood peak detection

Note, that the baseline model was not evaluated for flood events, since it always predicts a constant value, making it impossible to identify a peak value, which is necessary for this piece of analysis.

The metrics EPA and REP focus especially on the models' capability to correctly detect the flood peaks. For a better overview of the models' predictions in flood scenarios in general, each flood event and the corresponding predictions are visualized with the Python package matplotlib [31] and analysed.

The first flood event was the longest and lasted for 58 hours. The ARX model did not identify a flood at all, while the LSTM model managed to model the shape of the observed hydrograph well in the first approximately 40 hours. After the first 40 hours, it incorrectly predicted a second rise of the hydrograph, which surpassed the first rise in terms of magnitude. The COSERO model consistently overestimated the actual flow value but managed to simulate the hydrograph the best overall.

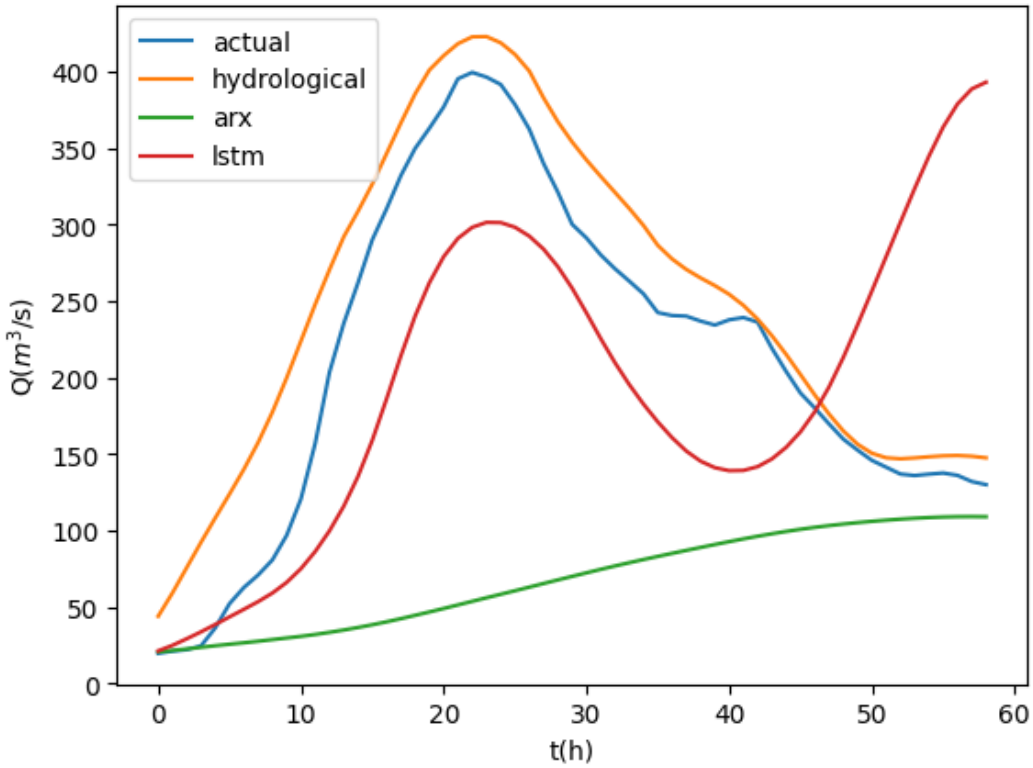


Figure 11: Hydrographs of the models' predictions and actual measurements of Flood 1 (15/05/2014 16:00 – 18/05/2014 02:00)

For the second flood, all models initially predicted the actual flow rate well, but both the ARX and the LSTM model failed to forecast the steep rise while the COSERO model made the most accurate prediction by far. After the drop, the LSTM actually had the most accurate predictions, but in general, it failed to properly model the rise and the maximum value, which is the most important thing in flood prediction. The ARX model barely predicted any rise in the flow value again.

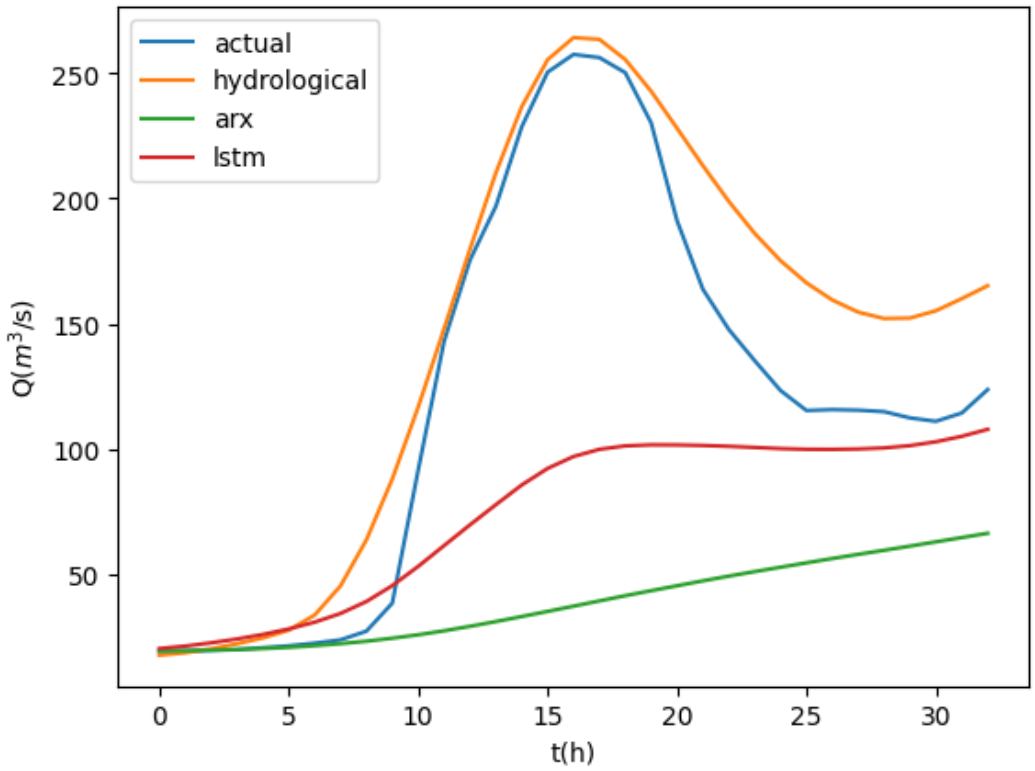


Figure 12: Hydrographs of the models' predictions and actual measurements of Flood 2 (27/05/2014 23:00 – 29/05/2014 07:00)

The third flood had a similar duration and magnitude as the second flood. The COSERO model initially overestimated the flow values by almost double the value, but in general did the best job again to predict the hydrograph correctly. The ARX model again did not recognise a huge jump in water flow. The LSTM model outperformed the ARX model but did also not get the magnitude and the peak correctly.

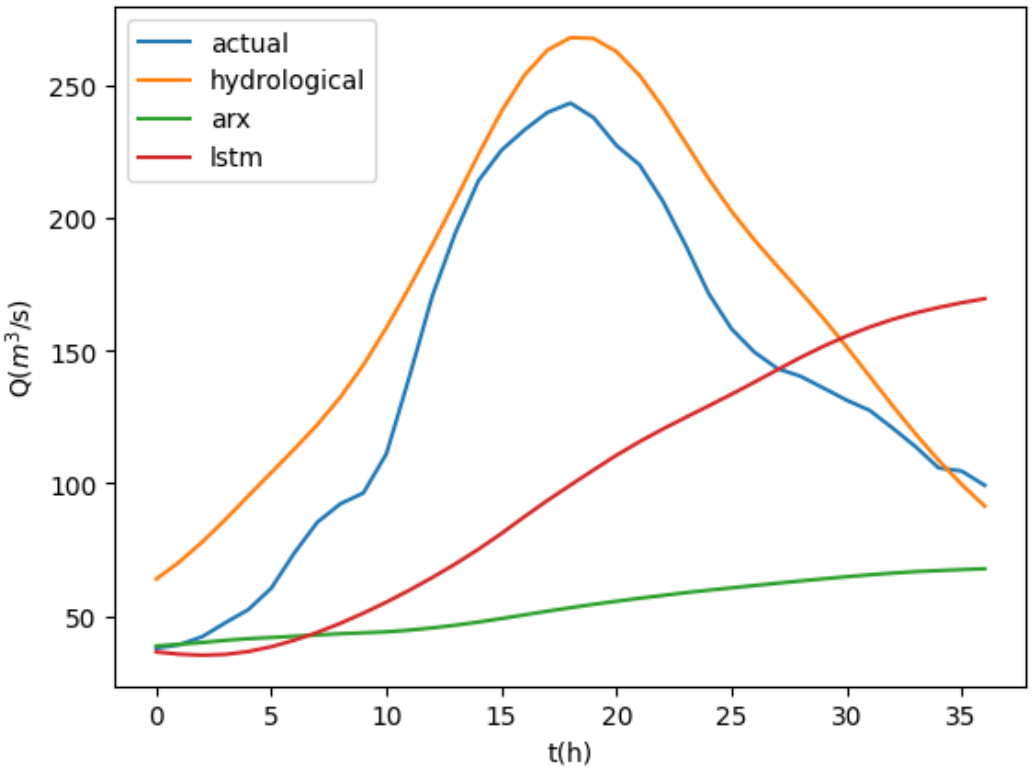


Figure 13: Hydrographs of the models' predictions and actual measurements of Flood 3 (23/10/2014 04:00 – 24/10/2014 16:00)

The fourth flood event was not as huge compared to the first three and also even shorter than flood 2 and 3. The COSERO model initially underestimated the flow values but managed to model the observed hydrograph well again. In this case, both ARX and LSTM did not forecast any significant rise in the hydrograph.

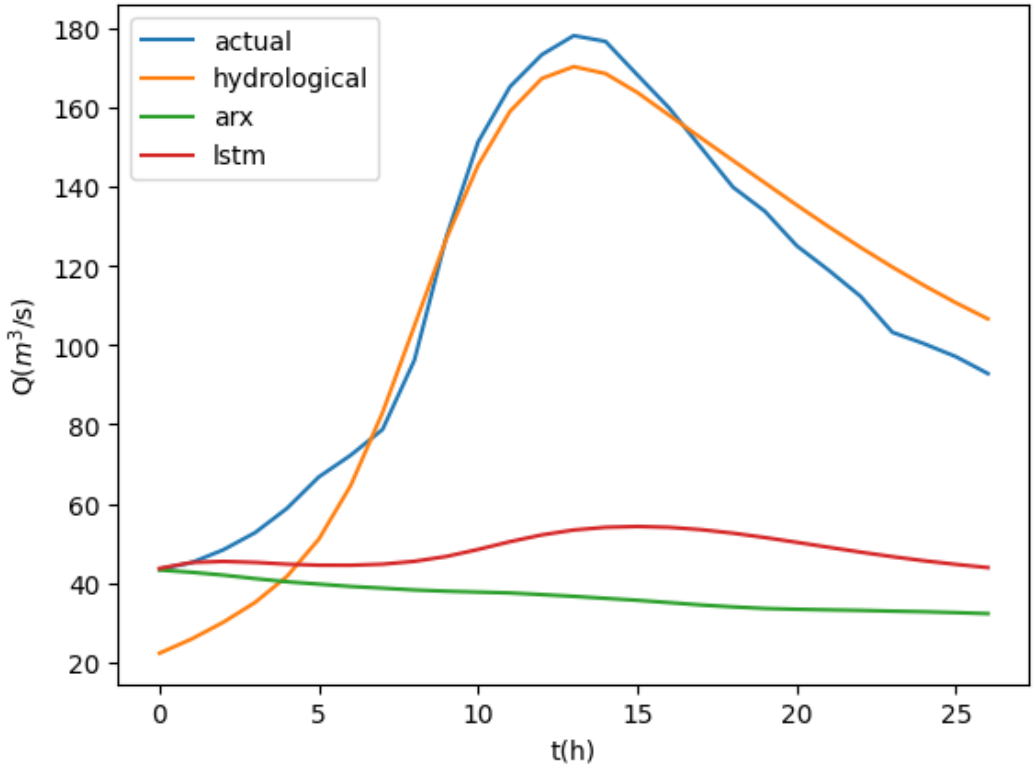


Figure 14: Hydrographs of the models' predictions and actual measurements of Flood 4 (10/01/2015 21:00 – 11/01/2015 23:00)

The last flood had similar characteristics as the fourth flood. This time, all three models had quite a significant deviation from the actual hydrograph. The ARX model and LSTM model both underestimated the actual flow value, while the COSERO model overestimated the actual flow value. However, since a flood is an event where it is better to overestimate the danger somewhat than to underestimate it, the prediction of the COSERO model is probably the best here.

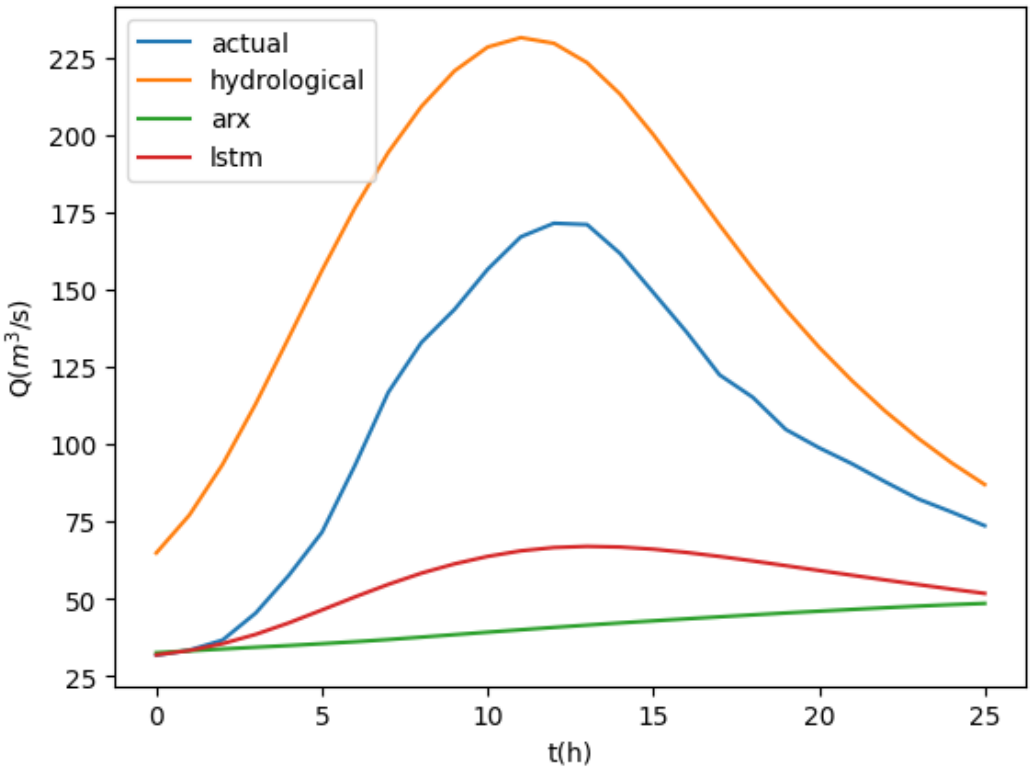


Figure 15: Hydrographs of the models' predictions and actual measurements of Flood 5 (01/02/2016 02:00 – 02/02/2016 03:00)

Overall, the COSERO model did the best job by far in accurately predicting flood events. The LSTM model most of the time somewhat got the shape of the hydrograph right, but constantly underestimated the magnitude of the flow value, while the ARX model did not even get the rough shape of the hydrograph correctly in all scenarios. One possible explanation, why the COSERO model performed so much better than the two ML approaches is that during its calibration flood events were of big interest. On the other hand, each training sample was treated equally for the ML models. Since there were many more non-flood- than flood-related training samples, this could explain the difference in performance.

6 Conclusion

This chapter aims to give an overview over all findings of this work, as well as limitations and further research questions that have arisen from this work.

6.1 Summary

Various approaches for predicting water flow rates or water levels with ML models have been made, as previously shown in the *Systematic Literature Review* chapter. Many different models were used in those works. In this work, two of those approaches were implemented: a linear, simple ARX model and the often in other works used, more complex LSTM model. The ML models were compared to a hydrological model, which is currently used by an official hydrological service (COSERO model).

6.1.1 Research Question Answer

The implemented ML models were compared to the COSERO model in terms of water flow prediction in general and in terms of prediction accuracy for flood events.

The general water flow forecasting capabilities were assessed for three different prediction horizons: 6, 12 and 24 hours. Both the ARX and the LSTM models were able to achieve almost the same performance scores as the COSERO model for the 6-hour prediction horizon. For the prediction horizons of 12 and especially 24 hours, the COSERO did not lose a lot of performance, while the ARX and the LSTM model were not able to achieve satisfactory results.

Looking at flood event prediction the COSERO model once again outperformed both the ARX and the LSTM model, both in terms of getting the right magnitude and timestamp of the flood peak as well as modelling the hydrograph during the flood period. While the LSTM model at least partially correctly detected flood events, the ARX model hardly predicted the correct rises and falls.

6.1.2 Further Findings

For the shorter prediction horizons of 6 and 12 hours, the simpler ARX model managed to outperform the LSTM model, even though the latter had significantly more parameters. On the other hand, the LSTM model achieved better results for the longer prediction horizon of 24 hours and delivered more accurate predictions for the flood events. This finding is consistent with the results found in the course of the literature research: Simpler models can provide better results for shorter prediction horizons, while more complex models are better suited for longer prediction horizons.

While training the ARX model was rather straightforward, training an appropriate LSTM model takes significantly more time. On the one hand, the training process in itself takes significantly longer, on the other hand, finding the optimal hyperparameter combination is a more difficult task due to far more hyperparameters to tune. Furthermore, many of the trained LSTM models were prone to initial small errors quickly adding up to very large errors.

Another significant aspect that should be considered is the explainability of models. The COSERO model represents the complete hydrological cycle in a simplified form, so hydrologists can detect an error in the model. For example, if the calculated soil moisture is incorrect and therefore the predicted flow is incorrect, it can be readjusted to minimise future errors. If an LSTM model predicts an incorrect flow value, it is not directly possible to understand where this error comes from, as the model is a black box. Here, additional Explainable AI methods would have to be used to gain more understanding and trust about the model predictions.

6.2 Limitations

The flow simulation of the hydrological model was carried out with actual measured precipitation and temperature values, thus measured precipitation and temperature data were also used as input data for the ML models. Since in practice calculations are made into the future, forecasts would be used as input here instead of measured data. It is possible that the results could change if one model can work better with possibly incorrect inputs than another.

In this work, a selection of two ML methods, namely ARX and LSTM, has been implemented. Even though the aim was to cover a wide range of approaches (traditional statistical and Deep Learning models), it was not possible to cover the full scope of possibilities. It is possible that different ML model would have led to different results.

Due to time constraints, it was not possible to conduct a fully comprehensive grid search for the LSTM model. Perhaps a different hyperparameter combination within or even outside of the hyperparameter ranges tested could have enhanced the LSTMs' model performance.

Furthermore, the experiment was conducted for one specific gauging station at one specific river. It is possible that the results would look different for rivers with other characteristics. Factors such as the size, length of the river, the size and nature in terms of hilliness, vegetation and soil of the catchment, as well as the general climate, could influence how well the flow value of a river can be modelled using ML. It is also possible that another type of water body would be better suited to modelling by an ML model, for example a lake.

6.3 Potential Further Research

The fact that not every ML model present could be implemented and evaluated in this work leaves a lot of room for potential further research in this field. Different approaches and models could have led to different results.

In this work, two black box ML models (i.e., which are solely based on input-to-output mapping) were investigated and tested against a purely hydrological model. This raises the question, how a hybrid model, which takes into account both approaches, ML and hydrological information, would perform compared to a purely hydrological model. Additional information, which could be used includes geographical and topological factors as well as flow values from gauging stations above the gauging station of interest.

The performance of the hydrological COSERO model was better than the two tested ML approaches in both general water flow forecasting and especially in terms of flood forecasting. The COSERO model was calibrated with a focus on flood events, which possibly was an advantage. This fact leads to the question if a different training strategy could have provided better results for the ML models. One possible approach could be to use some sort of sample weights, which means that samples where flood events occur get a higher weight than other samples. Another option would be to manually alter the training dataset in such a way that flood events take a higher proportion of the training dataset. One approach to achieve this could be to copy all flood-related training instances and apply data augmentation techniques. This could however come with the drawback of the overall water flow prediction performance declining.

In this work, the performance of the two ML models for a specific river (Erlauf) was analysed for a specific measuring point (Niederndorf). Future work could analyse whether the trends measured here are reflected in other water bodies. The COSERO model has its own snow module for areas where the temperature falls below zero degrees Celsius in winter, which calculates how much liquid is currently stored in the form of snow. It is possible that the difference between the COSERO model and the ML approaches would be smaller in areas where this circumstance would not come into play, or that the ML approaches could even achieve better results than the hydrological COSERO model. Also, the complex, mountainous topography in the upper reaches of the Erlauf was probably advantageous for the COSERO model, possibly in flatter areas an ML model would have better performance. All these questions could be explored in future works.

References

[1] N. W. Arnell and B. Lloyd-Hughes, “The global-scale impacts of climate change on water resources and flooding under new climate and socio-economic scenarios,” *Clim Change*, vol. 122, no. 1, pp. 127–140, 2014, doi: 10.1007/s10584-013-0948-4.

[2] Mohammad. Al-Fawa’Reh, A. Hawamdeh, R. Alrawashdeh, and M. T. Jafar, “Intelligent Methods for flood forecasting in Wadi al Wala, Jordan,” in *2021 International Congress of Advanced Technology and Engineering (ICOTEN)*, 2021, pp. 1–9. doi: 10.1109/ICOTEN52080.2021.9493425.

[3] R. Liu, C. Ye, P. Yang, Z. Miao, B. Liu, and Y. Chen, “Short-Term Prediction Model of Water Level Based on ATT-ConvLSTM,” in *2022 the 5th International Conference on Data Storage and Data Engineering*, in DSDE ’22. New York, NY, USA: Association for Computing Machinery, 2022, pp. 85–90. doi: 10.1145/3528114.3528128.

[4] “Lower Austria Water Level Forecasting Description.” https://www.noel.gv.at/wasserstand/#/de/Static/Erlauterungen/3 (accessed Apr. 13, 2023).

[5] “PRISMA Checklist.” http://prisma-statement.org/PRISMAstatement/checklist.aspx (accessed Apr. 13, 2023).

[6] “PRISMA Flow Chart.” http://prisma-statement.org/PRISMAStatement/FlowDiagram (accessed Apr. 13, 2023).

[7] K. Chaowanawatee and A. Heednacram, “Implementation of Cuckoo Search in RBF Neural Network for Flood Forecasting,” in *2012 Fourth International Conference on Computational Intelligence, Communication Systems and Networks*, 2012, pp. 22–26. doi: 10.1109/CICSyN.2012.15.

[8] A. R. Sanubari, P. D. Kusuma, and C. Setianingsih, “Flood Modelling and Prediction Using Artificial Neural Network,” in *2018 IEEE International Conference on Internet of Things and Intelligence System (IOTAIS)*, 2018, pp. 227–233. doi: 10.1109/IOTAIS.2018.8600869.

[9] K. Boukharouba, P. Roussel, G. Dreyfus, and A. Johannet, “Flash flood forecasting using Support Vector Regression: An event clustering based approach,” in *2013 IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2013, pp. 1–6. doi: 10.1109/MLSP.2013.6661958.

[10] S. Li, K. Ma, Z. Jin, and Y. Zhu, “A new flood forecasting model based on SVM and boosting learning algorithms,” in *2016 IEEE Congress on Evolutionary Computation (CEC)*, 2016, pp. 1343–1348. doi: 10.1109/CEC.2016.7743944.

[11] F. Liu, F. Xu, and S. Yang, “A Flood Forecasting Model Based on Deep Learning Algorithm via Integrating Stacked Autoencoders with BP Neural Network,” in *2017 IEEE Third International Conference on Multimedia Big Data (BigMM)*, 2017, pp. 58–61. doi: 10.1109/BigMM.2017.29.

[12] F. A. Ruslan, K. Haron, A. M. Samad, and R. Adnan, “Multiple Input Single Output (MISO) ARX and ARMAX model of flood prediction system: Case study Pahang,” in *2017 IEEE 13th International Colloquium on Signal Processing & its Applications (CSPA)*, 2017, pp. 179–184. doi: 10.1109/CSPA.2017.8064947.

[13] Y. Wu, Z. Liu, W. Xu, J. Feng, S. Palaiahnakote, and T. Lu, “Context-Aware Attention LSTM Network for Flood Prediction,” in *2018 24th International Conference on Pattern Recognition (ICPR)*, 2018, pp. 1301–1306. doi: 10.1109/ICPR.2018.8545385.

[14] Y. Ding, Y. Zhu, Y. Wu, F. Jun, and Z. Cheng, “Spatio-Temporal Attention LSTM Model for Flood Forecasting,” in *2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, 2019, pp. 458–465. doi: 10.1109/iThings/GreenCom/CPSCom/SmartData.2019.00095.

[15] T. Gohar, L. Hasan, G. M. Khan, and M. Mubashir, “Constraint Free Early Warning System for Flood Using Multivariate LSTM Network,” in *2022 2nd International Conference on Artificial Intelligence (ICAI)*, 2022, pp. 64–70. doi: 10.1109/ICAI55435.2022.9773495.

[16] D. Liu, W. Jiang, L. Mu, and S. Wang, “Streamflow Prediction Using Deep Learning Neural Network: Case Study of Yangtze River,” *IEEE Access*, vol. 8, pp. 90069–90086, 2020, doi: 10.1109/ACCESS.2020.2993874.

[17] G. Selva Jeba, P. Chitra, and U. M. Rajasekaran, “Time-series analysis and Flood Prediction using a Deep Learning Approach,” in *2022 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET)*, 2022, pp. 139–142. doi: 10.1109/WiSPNET54241.2022.9767102.

[18] S. Miau and W.-H. Hung, “River Flooding Forecasting and Anomaly Detection Based on Deep Learning,” *IEEE Access*, vol. 8, pp. 198384–198402, 2020, doi: 10.1109/ACCESS.2020.3034875.

[19] Y. Wu, W. Xu, J. Feng, S. Palaiahnakote, and T. Lu, “Local and Global Bayesian Network based Model for Flood Prediction,” in *2018 24th International Conference on Pattern Recognition (ICPR)*, 2018, pp. 225–230. doi: 10.1109/ICPR.2018.8546257.

[20] C. Karyotis, T. Maniak, F. Doctor, R. Iqbal, V. Palade, and R. Tang, “Deep Learning for Flood Forecasting and Monitoring in Urban Environments,” in *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, 2019, pp. 1392–1397. doi: 10.1109/ICMLA.2019.00227.

[21] Y. Wu, D. Yukai, and J. Feng, “Sparse Bayesian Flood Forecasting Model Based on SMOTEBoost,” in *2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, 2019, pp. 279–284. doi: 10.1109/iThings/GreenCom/CPSCom/SmartData.2019.00067.

[22] J. Feng, Z. Wang, Y. Wu, and Y. Xi, “Spatial and Temporal Aware Graph Convolutional Network for Flood Forecasting,” in *2021 International Joint Conference on Neural Networks (IJCNN)*, 2021, pp. 1–8. doi: 10.1109/IJCNN52387.2021.9533694.
[23] I. K. Nti *et al.*, “Enhancing Flood Prediction using Ensemble and Deep Learning Techniques,” in *2021 22nd International Arab Conference on Information Technology (ACIT)*, 2021, pp. 1–9. doi: 10.1109/ACIT53391.2021.9677084.
[24] M. Moishin, R. C. Deo, R. Prasad, N. Raj, and S. Abdulla, “Designing Deep-Based Learning Flood Forecast Model With ConvLSTM Hybrid Algorithm,” *IEEE Access*, vol. 9, pp. 50982–50993, 2021, doi: 10.1109/ACCESS.2021.3065939.
[25] R. Zhou and Y. Pan, “FloodDAN: Unsupervised Flood Forecasting based on Adversarial Domain Adaptation,” in *2022 IEEE 5th International Conference on Big Data and Artificial Intelligence (BDAI)*, 2022, pp. 6–12. doi: 10.1109/BDAI56143.2022.9862723.
[26] Y. Zhu, J. Feng, L. Yan, T. Guo, and X. Li, “Flood Prediction Using Rainfall-Flow Pattern in Data-Sparse Watersheds,” *IEEE Access*, vol. 8, pp. 39713–39724, 2020, doi: 10.1109/ACCESS.2020.2971264.
[27] “Austria-Forum Erlauf.” https://austria-forum.org/af/AEIOU/Erlauf,_Fluss (accessed Apr. 25, 2023).
[28] H. Kling, P. Stanzel, and C. Libisch, “Hochwasserprognose Erlauf Endbericht und Technische Dokumentation,” *[unpublished report and technical documentation]*. 2015.
[29] “WASSERSTANDSNACHRICHTEN und Hochwasserprognosen: Niederndorf - Durchfluss.” https://www.noel.gv.at/wasserstand/#/de/Messstellen/Details/207803/Durchfluss/3Tage (accessed May 16, 2023).
[30] “GeoSphere Austria.” https://www.geosphere.at/ (accessed Sep. 06, 2023).
[31] “Matplotlib.” https://matplotlib.org/ (accessed Aug. 18, 2023).
[32] “Afry.” https://afry.com/de-at (accessed Sep. 06, 2023).
[33] Kling H and Lerche F, “Evaluierung und Update COSERO Prognosemodelle,” *[unpublished report]*. 2023.
[34] “Python.” https://www.python.org/ (accessed Apr. 13, 2023).
[35] “Pandas.” https://pandas.pydata.org/ (accessed Apr. 13, 2023).
[36] “Rasterio.” https://rasterio.readthedocs.io/en/stable/ (accessed Apr. 13, 2023).
[37] “Numpy.” https://numpy.org/ (accessed Apr. 13, 2023).
[38] R. Pascanu, T. Mikolov, and Y. Bengio, “On the difficulty of training recurrent neural networks,” in *International conference on machine learning*, 2013, pp. 1310–1318.
[39] Y. Bengio, P. Frasconi, and P. Simard, “The problem of learning long-term dependencies in recurrent networks,” in *IEEE International Conference on Neural Networks*, 1993, pp. 1183–1188 vol.3. doi: 10.1109/ICNN.1993.298725.
[40] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput*, vol. 9, no. 8, pp. 1735–1780, 1997.

[41] I. G. Pechlivanidis, B. M. Jackson, N. R. Mcintyre, H. S. Wheater, and others, “Catchment scale hydrological modelling: a review of model types, calibration approaches and uncertainty analysis methods in the context of recent developments in technology and applications,” *Global NEST journal*, vol. 13, no. 3, pp. 193–214, 2011.
[42] G. K. Devia, B. P. Ganasri, and G. S. Dwarakish, “A review on hydrological models,” *Aquat Procedia*, vol. 4, pp. 1001–1007, 2015.
[43] H. Kling, P. Stanzel, M. Fuchs, and H.-P. Nachtnebel, “Performance of the COSERO precipitation–runoff model under non-stationary conditions in basins with different climates,” *Hydrological Sciences Journal*, vol. 60, no. 7–8, pp. 1374–1393, 2015, doi: 10.1080/02626667.2014.959956.
[44] “Scikit-learn.” https://scikit-learn.org/stable/ (accessed Apr. 13, 2023).
[45] “Skforecast.” https://skforecast.org/0.8.1/index.html (accessed Jul. 20, 2023).
[46] “Keras.” https://keras.io/ (accessed Jul. 20, 2023).
[47] “Scipy.” https://scipy.org/ (accessed Aug. 23, 2023).

List of Figures

List of Tables