Flood prediction with ANN

Alicia Bayerl, Rosanna Beckmann, Tim Kapferer, Michael Ramich,
Hendrik Schmidt, Maximilian Springer

July 24, 2022

Abstract

Flood prediction is crucial for reducing human and economic damage. This paper explores the prediction of floods with Long Short-Term Memory (LSTM) artificial neural networks. The input data consists of water level and weather data including wind, precipitation, air pressure, moisture, and temperature from several weather stations in the area of Hamburg, Germany. The Deutscher Wetterdienst and the Hamburg Port Authority provided the data sets. Results indicate that simple LSTM-based models are able to predict floods. For actual flood prediction, our data limits proper results, even though LSTMs are able to predict extreme events from a relatively small amount of data.

1 Introduction

Climate change makes heatwaves, droughts, and heavy rainfalls more likely. The latter can induce flooding events as happened in July of 2021 in parts of northern Europe. While it is not possible to attribute certain extreme weather events to climate change, it is possible to determine whether climate change makes the preconditions for such events more likely. Climate change has a two-fold impact in that it on the one hand increased the intensity of the rainfall event by 3 - 19% and on the other hand increased the likelihood of such an event happening by a factor of 1.2 - 9 [9].
Physically-based models are well adapted to predict various types of flooding scenarios but to do so they rely on data sets that require high amounts of computation which makes them unsuitable for short-term prediction [12]. Furthermore, the creation of physically-based models often requires expert knowledge of the hydrological parameters involved [8].
Given that we do not have any expert knowledge of the hydrological components involved in predicting floods we would like to make use of ANNs to predict flooding events. ANNs are suited for this task because they can capture the non-linearity underlying flooding events [12].

1.1 Objectives

Weather is a highly complex dynamical system depending on many parameters and non-linear functions. Weather can cause floods, which can be destructive and catastrophic. In this paper we would therefore answer the following research questions: Is it possible to set up and train an Artificial Neural Network (ANN) to accurately predict flooding in a geographically limited area? In setting up the ANN we hope to gain insights into which architectures might work best, which parameters are most important, and if there are any other shortcomings ANNs might have. In doing this we will focus on the Long Short Term Memory networks (LSTMs).

1.2 Related work

The current flood predictions for Hamburg are conducted by the Hamburger Sturmflutwarndienst (WADI). The WADI uses statistical models based on decade-long observations of flooding events to produce half-hourly predictions [13]. They base their predictions on the water level as well as weather data [13]. While machine learning, as well as deep learning techniques, have been applied to forecast floods in other regions of the world [5], in the best of our knowledge no major efforts were undertaken in applying deep learning techniques to enhance the forecasts ability for Hamburg. One project [4] tried

to improve short-term precipitation forecast to better predict flooding events. They used an already existing hydrological model but used preprocessed radar images as input data [4]. Xuan-Hien Le et al. used an LSTM model to predict the flow rate at the Da River basin in Vietnam. Their main input variables were the daily flow rate and precipitation data [11]. Their one-day forecasting model reached a 99% Nash–Sutcliffe efficiency [11]. Zaharaddeen Karami Lawa et al. compared the predictive power of decision trees, logistic regression, and support vector classification for flood prediction in Kebbi state Nigeria [10]. As input data, they used rainfall data. The logistic regression model performed best among the three compared [10].

1.3 Contributions

Hendrik Schmidt, Maximilian Springer, Michael Ramich, and Tim Kapferer worked on the data processing and the implementation of the LSTM models. Alicia Bayerl and Rosanna Beckmann summarized current literature and created the poster. All authors were involved in writing the paper.

2 Methods

For the loading and preprocessing of the data, we used Pandas and NumPy. The models were built using the TensorFlow framework as well as Keras. All of our code can be found in this GitHub repository.

2.1 Data

Our data were provided by the Deutscher Wetterdienst (DWD) as well as the Hamburg Port Authority (HPA). From the DWD we gathered our weather data and from the HPA we gathered water level data.

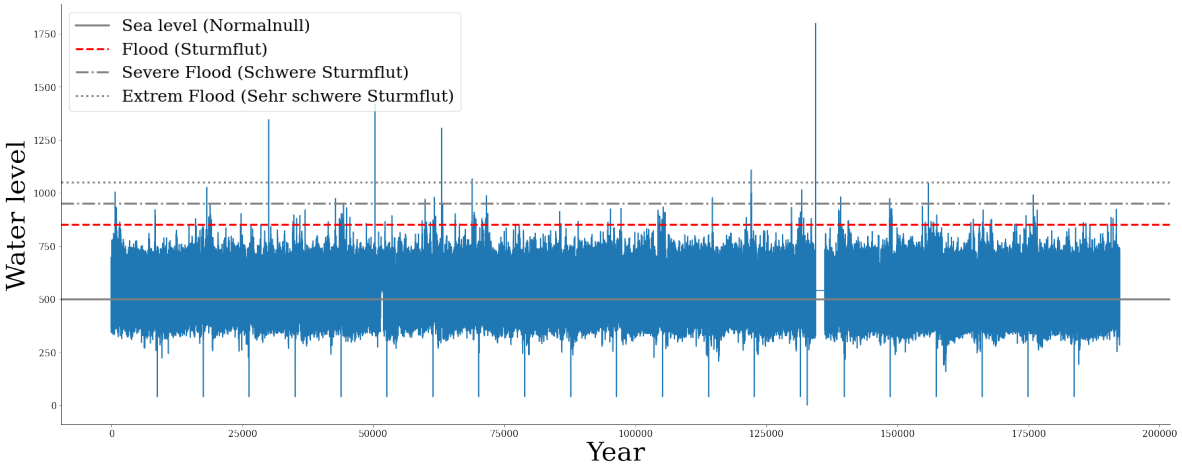

Figure 1: Water level of Hamburg, St.Pauli from 2000 to 2020.

From the HPA we got data for the water level in a 10-minute resolution ranging from 1950 to 1992 for the monitoring stations in Schöpfstelle, St. Pauli, Harburg, and Seemannshöft. All of them are located in Hamburg. We also downloaded water level data for St. Pauli from their web portal in a 1-minute resolution which covered the time span from 2000 to 2020. We accessed the weather data from DWD via their online portal. We downloaded hourly data about wind speed, wind direction, precipitation, air pressure and moisture for the weather stations in Cuxhaven, Freiburg/Elbe, Mittelnkirchen-Hohenfelde, Hamburg Neuwiedenthal, Hamburg Flughafen, and Hamburg St. Pauli from 2000 to 2020. Including locations around Hamburg may enable the model to recognize weather changes. Especially including Cuxhaven may allow capturing more information about the ocean's weather conditions, which can have a crucial impact on floods in Hamburg. The locations of the individual weather stations are depicted in Figure 2.

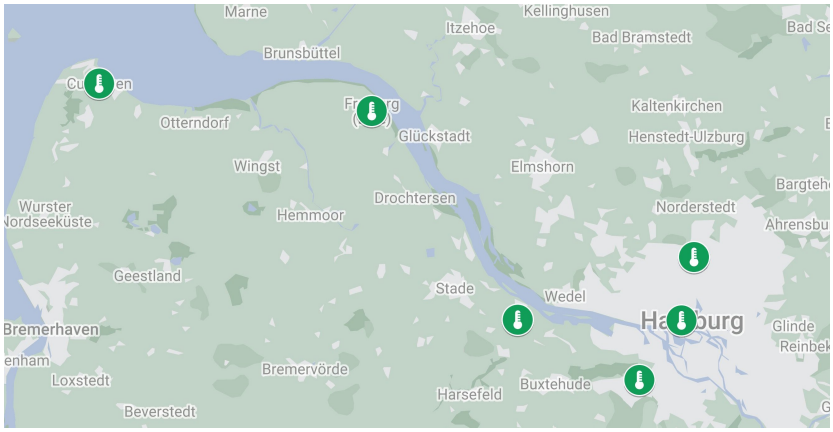

Figure 2: Visualisation of the weather stations' locations.

2.1.1 Preprocessing

We preprocessed both the weather and the water level data in similar ways. We replaced missing values in the dataset by using NumPy's interpolate function. In the case of the hourly water level data, we dropped the year 2016 because it had missing data for more than three consecutive months. The wind direction in the weather data set was given in degrees, which we transformed into radian and then into x and y components. This enables the model to better capture the direction of the wind. We also added information about the time of day and time of year to the data sets. This was done by converting the time of year and time of day to cosine as well as sine curves. The same time in different years will thus be encoded by the same value, allowing our model to better capture seasonality trends. The ratio of flooding events to non-flooding events was 1 to 345 after all of the preprocessing steps. This scarcity of flooding events presented challenges for training the model.

2.1.2 Constructing the datasets

From the data, we started building a daily dataset but soon realized, that this dataset would, due to its low temporal resolution, probably not provide enough information to our models to learn the underlying structure of flooding events. Thus, we only focused on building a dataset containing all of our data in hourly resolution.

Weather data availability					
Station name	Wind	Precipitation	Air pressure	Moisture	Temperature
Cuxhaven	X	X	X	X	X
Freiburg/Elbe	X			X	
Mittelnkirchen		X		X	X
Neuwiedenthal		X		X	X
Flughafen	X	X	X	X	X
St. Pauli	X				

Table 1: Weather data availability at the weather stations.

In Table 1 one of the main challenges in creating the datasets is visible. Most of the data is missing at two stations. The air pressure data is only existing for the Cuxhaven and the Hamburg Flughafen station. We decided to keep this data as it, especially in the case of Cuxhaven, holds relevant information about whether a storm is happening.
As a target for the model to predict we chose the water level in St. Pauli depicted in Figure 1. The two main reasons for this were that we had the most detailed information about St. Pauli and that it is located in the city, hence, a flooding event would affect many people. We arrived at the final dataset by joining the water level and the weather data. We thus arrived at a dataset containing hourly values

for 24 features from the year 2000 to 2020.

2.2 The models

Neural networks are versatile function approximators. However, traditional neural networks are not well suited to model long-term dependencies. Recurrent neural networks (RNNs) can do this, but only in a close context. They are not that well suited to model long-term dependencies in unison with the context[15]. In contrast, LSTMs can do both [14]. Flooding events are dependent on the current weather conditions, but we can only meaningfully determine whether certain conditions constitute the preconditions for a flood by taking previous flooding events and their preconditions into account. LSTMs are able to model these dependencies because they have an additional cell state that stores information from previous time steps and can be modified in each time step[2].

2.2.1 The One Layer LSTM model

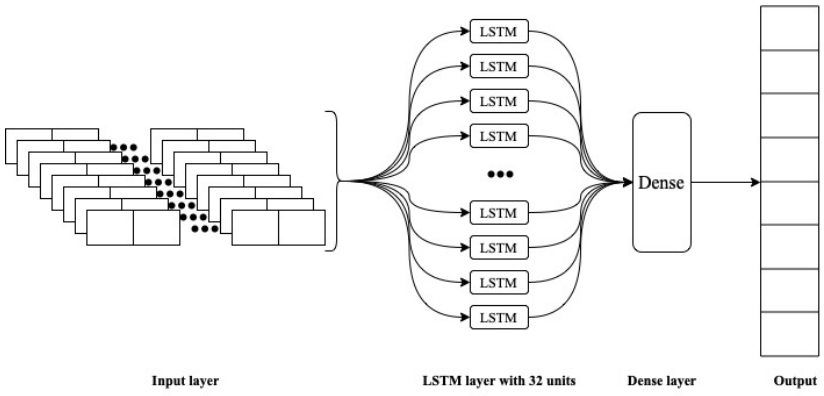

Figure 3: Architecture of the one layer LSTM.

The One Layer LSTM model gets 48 hours of input data to predict the water level for the next 12 days. The input data contains all 24 features from the dataset we created. Figure 3 depicts the One Layer LSTM's architecture for the simplified case of eight hours of input data leading to eight hours of prediction. The input is fed into an LSTM layer containing 32 units which then feeds into a dense layer to produce the predictions.

2.2.2 The Basic LSTM model

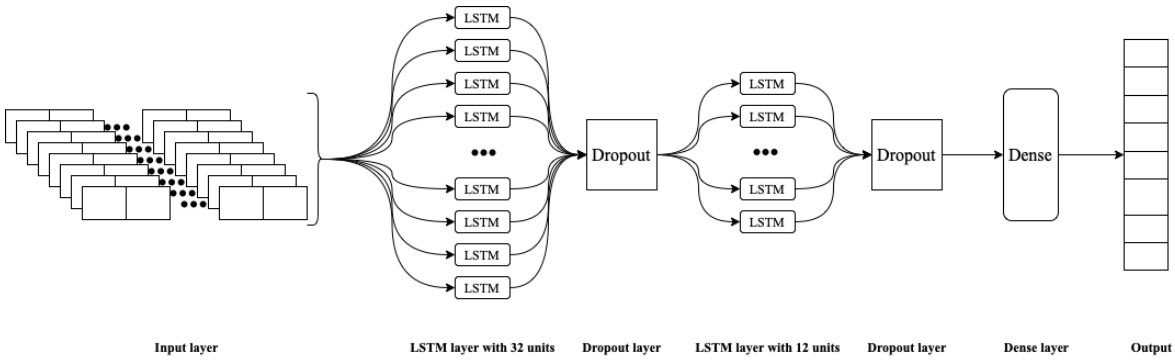

Figure 4: Architecture of the basic LSTM model.

The architecture of the Basic LSTM model is depicted in Figure 4. As input we use 48 hours of consecutive time series data, holding our 24 features. We used varying degrees of dropout, ranging from none to 0.3. The first LSTM layer is made up of 32 units while the second only is made up of 12 units. The dense layer predicts the water level for the next 12 days.

2.2.3 The Feedback LSTM model

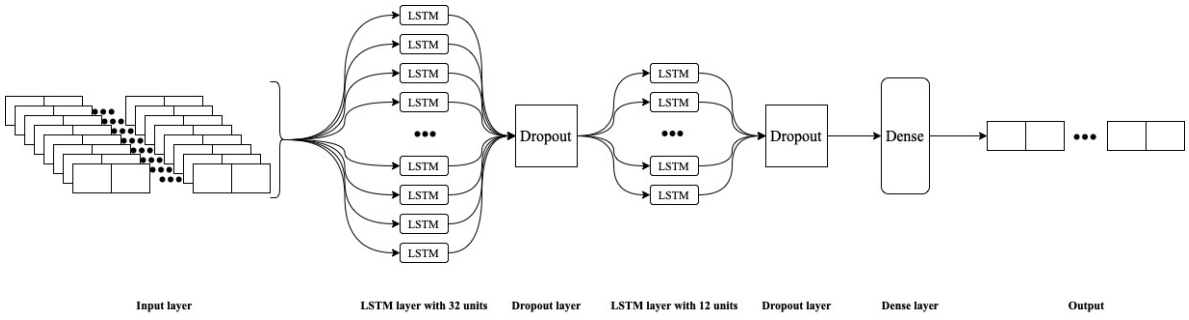

Figure 5: Architecture of the Feedback LSTM model.

The architecture of the Feedback LSTM model is depicted in Figure 5. This model has the same underlying architecture as the Basic LSTM model. In contrast to the LSTM model above this model's dense layer predicts all features for one-time step. To arrive at an output of 12-time steps, we feed the predicted time step back into the model as input. This is repeated until we arrive at the desired output sequence length. The whole process is depicted for an input with two features and an output sequence length of 7 in Figure 6.

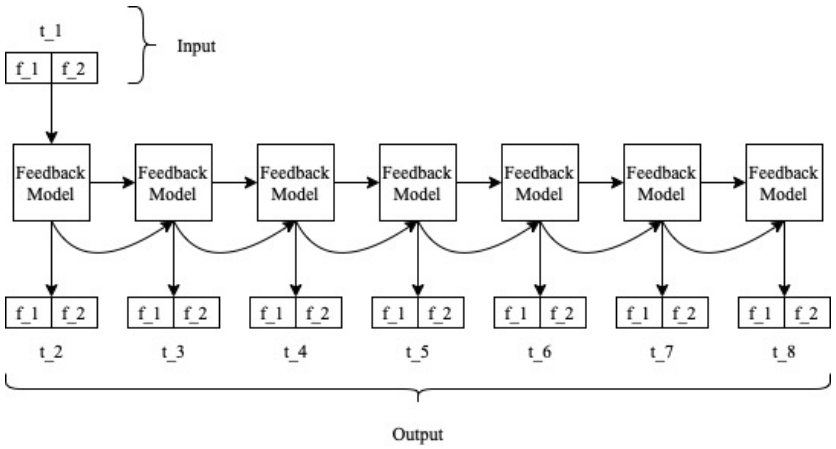

Figure 6: Simplified visualisation of how the Feedback LSTM predicts future values.

2.2.4 Loss functions

In the beginning, we trained all of our models using the mean-squared error (MSE) as the loss function. This allowed the models to capture the trend of the water level but seemed to work not as well for predicting the flooding events.
We think that this is due to the imbalance of the data since flooding events are much less likely, thus the MSE can be low even if the model underestimates the water level. We, therefore, set up a custom loss function. This loss function took as a baseline the MSE but added an extra penalty for every flooding event that was not predicted by the model. As a value for the penalty, we used the ratio of flooding events to non-flooding events. This allows us to address the data imbalance without resorting to over- or undersampling our data[1].

3 Results

All of the models above were trained for 20 epochs with Adam and the default learning rate. For evaluation, we used the Mean Absolute Error (MAE) and the Root Mean Squared Error (RMSE). We

Loss	Model	MAE	RMSE	Detected Floods
MSE Loss	One Layer LSTM	0.139	0.241	281 (61.62%)
	Basic LSTM	0.138	0.23	243 (53.29%)
	Feedback LSTM	0.131	0.221	58 (12.72%)
Custom Loss	One Layer LSTM	0.136	0.239	323 (70.83%)
	Basic LSTM	0.142	0.24	239 (52.41%)
	Feedback LSTM	0.132	0.23	187 (41.01%)

Table 2: Results of the different models in terms of MAE, RMSE and the number of detected floods

also set up a dataset that contained all the floods at each hour during a 12-hour period preceded by 48 hours of data. The floods were split according to whether they occurred in the train, validation, or test dataset. We only used the test dataset to evaluate whether a model is able to predict unseen flooding events. This dataset contains 456 separate flooding events. The results are displayed in Table 2. Both the One Layer and Basic LSTM perform equally in terms of MAE and RMSE when trained with the MSE loss and with the Custom loss. The One Layer LSTM detects more floods than the Basic LSTM and the Feedback LSTM for both loss functions. Results indicate that training with the custom loss function detects more floods in the case of the One Layer LSTM and the Feedback LSTM. For the basic LSTM, the MSE and the Custom loss function performance are equal. The Feedback LSTM improves a lot using the custom loss function, however, it still performs poorly compared to the One Layer LSTM and the Basic LSTM model.

4 Discussion

The data consists of local weather conditions with short temporal resolution. Thus the data might not represent weather changes and the model fails long-term prediction. Extending the spatial resolution of the data can increase the temporal resolution by further including the weather around the target area, as we did by adding locations further off Hamburg. Improving on that, long-term weather predictions and satellite images might help to increase the temporal resolution [7]. The former - if available - can be simply added to the current data set, whereas the latter requires further processing of the images. Furthermore, the imbalance of the data is a problem. The likelihood of a flood is low, thus the mean squared error loss function can be low even if no flood got correctly predicted. Simply close-to-average predictions might be the best mostly. This however results in a model prone to false negative predictions, which can be catastrophic in the case of floods. We introduced a custom loss function to increase the error largely when the model did not predict the water level of the flood. However further steps could be taken to balance the data using resampling methods, this might require a dataset reaching further back in time to include more floods.
The custom loss function and resampling methods both achieve the same goal of increasing the probability that the model will put more weight on the extreme samples. There is, however, a fine line where increasing the frequency or the weight of the extreme samples leads to more inaccurate predictions for the normal samples[3]. This may lead to the model predicting floods where none occur. One could implement different loss functions such as the EVL loss [3] which put more emphasis on the training dataset's statistical properties to avoid such problems. Other studies [6] tried to tackle this problem by introducing more weather parameters. But the model still underestimated the severeness of the extreme events [6].
Currently, we only tested the model for Hamburg but theoretically, it could be applied to other locations if identical data conditions are available. However, it might be required to train the model again to learn the dynamics of the location, as there might be other causes of floods at different locations.

5 Conclusion

We were able to predict some floods with our models. The One Layer LSTM performed the best when trained with the Custom Loss, with 323 predicted floods. Our loss function enhanced the basic LSTM model's predictive power. While this shows that LSTMs are able to predict extreme events from relatively sparse data, this still is too little to be applied for real flood prediction. One major

limitation might be our data. The model used by the WADI includes more data from the sea ranging to Great Britain. Including this data might enhance the predictive power even further.

References

[1] Azim Ahmadzadeh et al. "Rare-Event Time Series Prediction: A Case Study of Solar Flare Forecasting". In: *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*. 2019, pp. 1814–1820. DOI: 10.1109/ICMLA.2019.00293.

[2] Francois Chollet. *Deep learning with Python*. Simon and Schuster, 2021.

[3] Daizong Ding et al. "Modeling extreme events in time series prediction". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2019, pp. 1114–1122.

[4] Thomas Einfalt, Sandra Hellmers, and Alrun Jasper-Tönnies. "Urban flood prevention based on ensemble precipitation forecasts". In: *NOVATECH 2019*. 2019, pp. 1–4. ISBN: 978-2-917199-09-1. DOI: 10.15480/882.3455. URL: http://hdl.handle.net/11420/5054.

[5] Parag Ghorpade et al. "Flood Forecasting Using Machine Learning: A Review". In: *2021 8th International Conference on Smart Computing and Communications (ICSCC)*. IEEE, 2021, pp. 32–36. ISBN: 978-1-7281-9687-9. DOI: 10.1109/ICSCC51209.2021.9528099.

[6] M. Ionita et al. "Predicting the June 2013 European Flooding Based on Precipitation, Soil Moisture, and Sea Level Pressure". In: *Journal of Hydrometeorology* 16.2 (2015), pp. 598–614. DOI: 10.1175/JHM-D-14-0156.1. URL: https://journals.ametsoc.org/view/journals/hydr/16/2/jhm-d-14-0156_1.xml.

[7] Sharad Kumar Jain et al. "A Brief review of flood forecasting techniques and their applications". In: *International Journal of River Basin Management* 16.3 (2018), pp. 329–344.

[8] Byunghyun Kim et al. "Urban flood modeling with porous shallow-water equations: A case study of model errors in the presence of anisotropic porosity". In: *Journal of Hydrology* 523 (2015), pp. 680–692.

[9] Frank Kreienkamp et al. "Rapid attribution of heavy rainfall events leading to the severe flooding in Western Europe during July 2021". In: *World Weather Attribution* (2021).

[10] Zaharaddeen Karami Lawal, Hayati Yassin, and Rufai Yusuf Zakari. "Flood Prediction Using Machine Learning Models: A Case Study of Kebbi State Nigeria". In: *2021 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)*. IEEE, 2021, pp. 1–6. ISBN: 978-1-6654-9552-3. DOI: 10.1109/CSDE53843.2021.9718497.

[11] Xuan-Hien Le et al. "Application of long short-term memory (LSTM) neural network for flood forecasting". In: *Water* 11.7 (2019), p. 1387.

[12] Amir Mosavi, Pinar Ozturk, and Kwok-wing Chau. "Flood prediction using machine learning models: Literature review". In: *Water* 10.11 (2018), p. 1536.

[13] Sylvin H. Müller-Navarra et al. "Sturmflutvorhersage für Hamburg 1962 und heute". In: (2012).

[14] Christopher Olah. *Understanding LSTM Networks*. 2015. URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[15] Ralf C Staudemeyer and Eric Rothstein Morris. "Understanding LSTM–a tutorial into long short-term memory recurrent neural networks". In: *arXiv preprint arXiv:1909.09586* (2019).