water

MDPI

Article

A Spatiotemporal Deep Learning Approach for Urban Pluvial Flood Forecasting with Multi-Source Data

Benjamin Burrichter [1,*], Julian Hofmann [2], Juliana Koltermann da Silva [1], Andre Niemann [3] and Markus Quirmbach [1]

[1] Institute of Civil Engineering, University of Applied Sciences Ruhr West, 45479 Mülheim an der Ruhr, Germany; juliana.koltermanndasilva@hs-ruhrwest.de (J.K.d.S.); markus.quirmbach@hs-ruhrwest.de (M.Q.)
[2] Institute for Hydraulic Engineering and Water Resources Management, RWTH Aachen University, 57074 Aachen, Germany; hofmann@iww.rwth-aachen.de
[3] Institute for Hydraulic Engineering and Water Resources Management, University of Duisburg-Essen, 45141 Essen, Germany; andre.niemann@uni-due.de
* Correspondence: benjamin.burrichter@hs-ruhrwest.de

Abstract: This study presents a deep-learning-based forecast model for spatial and temporal prediction of pluvial flooding. The developed model can produce the flooding situation for the upcoming time steps as a sequence of flooding maps. Thus, a dynamic overview of the forthcoming flooding situation is generated to support the decision of crisis management actors. The influence of different input data, data formats, and model setups on the prediction results was investigated. Data from multiple sources were considered as follows: precipitation information, spatial information, and an overflow forecast. In addition, models with different layers and network architectures such as convolutional layers, graph convolutional layers, or generative adversarial networks (GANs) were considered and evaluated. The data required to train and test the models were generated using a coupled hydrodynamic 1D/2D model. The model setup with the inclusion of all available input variables and an architecture with graph convolutional layers presented, in general, the best results in terms of root mean square error (RMSE) and critical success index (CSI). The prediction results of the final model showed a high agreement with the simulation results of the hydrodynamic model, with drastic reductions in computation time, making this model suitable for integration into an early warning system for pluvial flooding.

Keywords: urban pluvial flooding; deep learning; spatiotemporal modeling; real-time flood forecasting

Citation: Burrichter, B.; Hofmann, J.; Koltermann da Silva, J.; Niemann, A.; Quirmbach, M. A Spatiotemporal Deep Learning Approach for Urban Pluvial Flood Forecasting with Multi-Source Data. *Water* **2023**, *15*, 1760. https://doi.org/10.3390/w15091760

Academic Editor: Gonzalo Astray

Received: 29 March 2023
Revised: 24 April 2023
Accepted: 27 April 2023
Published: 3 May 2023

1. Introduction

Pluvial flooding caused by heavy rainfall poses a high safety risk; for highly sealed urban areas in particular, precipitation becomes almost exclusively runoff. According to the sixth report of the Intergovernmental Panel on Climate Change (IPCC) [1], the number and intensity of heavy rainfall events have increased in recent years. This trend will likely persist because of global warming. In combination with the ongoing urbanization of cities, an increase in the frequency of pluvial flood events and in the resulting risk is expected in the future [2]. Since pluvial flooding, compared to fluvial flooding, can theoretically occur anytime and anywhere in urban catchments, comprehensive protection is not possible from a technical and economic point of view. Thus, early warning systems are essential to enable proactive protection in an incident. In addition, actors in municipal crisis management, as potential users of real-time warning systems, have high expectations of the reliability of the warning alerts. Spatially and temporally precise predictions are required for efficient action in a crisis and to avoid wrong decisions as far as possible. In summary, there are two competing requirements listed by Zhao et al. [3] for predictive models used in real-time warning systems:

- High temporal and spatial resolution of flood forecasts;
- Sufficient lead time between prediction and event occurrence.

Hydrodynamic (HD) computational models are a widely used tool for spatiotemporal high-resolution modeling of pluvial flood events. Initially, the main focus was on modeling the sewer network, but in recent years modeling the surface flooding process has become increasingly essential, and coupled modeling of both systems has become state of the art. The outputs are high-resolution 2D water level maps showing the flood hazard. Various studies on the validation of simulation results using images from social networks [4,5], surveillance camera footage [6], or reported insurance claims [7] have shown the excellent quality of these models. However, this quality is accompanied by high computational costs [8], which means that computational times can quickly reach several hours to days for a single event, depending on the study area size. Compared to flash floods in natural watersheds, which are triggered by advective precipitation events and can be predicted with numerical weather models for multiple hours or several days [9], pluvial flash floods are usually caused by convective precipitation cells. With current nowcasting models, these events can only be predicted with lead times of up to two hours [10–12]. Due to the short lead times, hydrodynamic models are presently unsuitable for real-time usage because of their long computation times. Therefore, the field of application is limited to the simulation of individual scenarios to identify general flooding hazards.

To meet the second requirement of sufficient lead time, many approaches to developing real-time warning systems for pluvial flooding focus on minimizing the computation times of hydrodynamic computational models as far as possible. For this purpose, the level of detail of the models can be reduced by considering only hotspots or reducing the resolution of the computational network [13]. Both approaches were combined by Hofmann and Schüttrumpf [14] for application in a real-time warning system. Other methods focus on increasing computational speed through parallel data processing [15,16] or simplifying computational operations [3,17,18]. However, reducing the level of detail or simplifying computational processes is accompanied by reduced accuracy of the computational results.

Other investigated approaches to reducing computation times use data-driven models to estimate flood extends. Most of these approaches are based on machine learning and often on deep learning. Especially, deep learning has recently achieved great success in the field of image processing [19–22]. The developed methods are applied in more and more areas such as earth system sciences [23] or water resources management [24]. In terms of flood modeling, Mosavi et al. [25] provided a general overview of existing machine learning models and Bentivoglio et al. [26] reviewed deep learning models. These models are usually trained with the results of hydrodynamic models to achieve similar results in a fraction of the time. Some of the investigated approaches differ significantly concerning the considered methods. For example, Bermúdez et al. [27] used an artificial neural network (ANN) to predict the maximum overflow volumes in subcatchments for a precipitation event. Depending on the predicted overflow volumes, a suitable flooding situation is selected from a result catalog of pre-simulated events. A similar approach was taken by Jhong et al. [28] and Lin et al. [29] with a support vector regression (SVR)-based model. They first used an SVR model to predict the water level hydrograph on the ground surface at various reference points in the study area. Subsequently, an SVM model was used to determine the inundation areas depending on the predicted water level at these points in combination with geographic information.

Bermúdez et al. [30] also developed an SVR model to predict floodplains. However, the prediction of water levels is performed directly for 25,000 points instead of only for a few reference points so that spatial information about the flooding event is instantly available. Berkhahn et al. [31] applied the same approach, where the method used is an ensemble of fully connected multi-layer perceptron layers. However, a disadvantage of fully connected networks is the high number of weights (parameters) to be trained and the associated high computational and memory requirements. This differs from convolutional neural networks (CNNs), which share the weight matrix in space, significantly reducing the number of

weights [32]. Guo et al. [33] used CNNs in an autoencoder architecture and trained a model to compute the flooding situation for entire cities. Hofmann and Schüttrumpf [34] also adopted CNNs but organized them in a generative adversarial network (GAN). To enable the transferability of the trained models to other areas, Guo et al. [33] and Löwe et al. [35] utilized spatial information as an additional input variable. Seleem et al. [36] also took advantage of spatial inputs and highlighted the good performance of a CNN architecture over a random forest algorithm regarding transferability. In addition, do Lago et al. [37] used spatial information in combination with a GAN to distribute the rainfall-runoff volume determined by a hydrologic model in a study area. Moreover, deep learning models have shown promising results in similar tasks such as flood sensitivity modeling [38–40] or fluvial flooding prediction [41,42].

In this work, a deep learning model for temporal and spatial prediction of pluvial flooding is developed for integration into an early warning system. The target variable represents a sequence of flood grids with a 2 m × 2 m resolution for the forecast horizon. Three potential variables are considered as inputs whose effect on the prediction quality is investigated as follows: (i) the fallen, as well as the predicted precipitation; (ii) spatial information on terrain properties and degree of pavement; (iii) the predicted overflow hydrographs for the forecast horizon. The investigations aim to check which inputs are required and how they must be provided to the model. The main contributions of this study can be summarized as follows:

1. Development of a prediction model for pluvial flooding based on deep learning that can predict the spatial and temporal evolution of the flooding situation. In contrast to other studies investigating the use of deep learning to predict pluvial flooding [31,33–35], the model output is a flooding sequence for the upcoming time steps instead of the maximum water levels. The chosen model design also allows predictions to be generated at any point in an event and is not limited to specific durations of an event. The accuracy of the results is expected to be as close as possible to that of physically based models, with drastically reduced computation times at the same time.
2. Compared to existing studies on predicting pluvial flash floods using deep learning approaches [31,33–35], the sewer network is considered as an extra retention volume here. To achieve this, an event-specific overflow forecast is taken as an additional input variable informing whether the sewer network is overloaded or not. In subsequent operational use, this input can be provided either by hydrodynamic sewer network models or data-driven models.
3. Different model setups are evaluated. This refers, on the one hand, to the considered model inputs and in the case of overflow prediction, to the data format and the model architecture depending on it. Furthermore, different modern deep learning architectures such as encoder-decoder networks, graph neural networks, or generative adversarial networks are combined and compared with each other in the investigations.

2. Methodology

2.1. Modeling Concept

The overall model structure is shown in Figure 1. The model aims to calculate the upcoming flooding situation for the next time steps starting from a time of observation. For this purpose, different precipitation information, spatial information, and an overflow forecast are available as potential input. Together with the predicted inundation areas, three different data formats are thus considered as follows:

- 1D time series (precipitation information and overflow forecast): These are time series whose values vary along the temporal axis but are assumed to be constant over the spatial extent of the study area (precipitation) or correspond only to a single spatial unit in the study area (overflow).
- 2D raster (spatial information): These are raster data sets whose values vary across the spatial extent of the study area but are assumed to be constant over time.

- 3D raster sequence (predicted inundation areas): These are grid sequences with the same format as video sequences. The values vary both spatially and temporally.

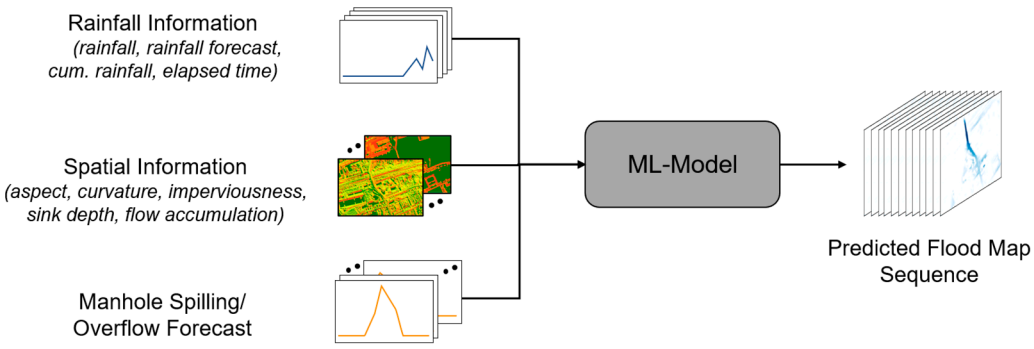

Figure 1. Model setup with all potential inputs (**left**) and the target variable (**right**).

The consideration of different data types is a special requirement for development of the machine learning (ML) model, which has to be capable of processing them together. This condition severely limits the number of suitable ML methods. Furthermore, the methods must be able to efficiently process image data (2D raster) and especially video data (3D raster sequence) and recognize structures within them. For this reason, the focus here is on artificial neural networks, which have proven to be particularly efficient in similar problems such as precipitation nowcasting (e.g., [43–45]) or various traffic forecasting tasks (e.g., [46–48]). Figure 1 presents the proposed model setup with the potential inputs and the predicted flooding situation as the target.

2.2. Considered Layers and Network Architectures

2.2.1. Fully Connected Layers

In artificial neural networks with fully connected layers, all neurons of one layer are fully connected to the neurons of the following layer. A widely used network architecture with fully connected layers is the multilayer perceptron (MLP), based on the perceptron developed by Rosenblatt [49]. This is a mathematical model for information processing that receives input, weights it, sums it up, and passes it on, according to an activation function. MLPs consist of multiple perceptrons organized in fully connected layers. The network output depends on the weights between the layers. These must be adjusted in a training process using an optimization algorithm to minimize the error between outputs and targets. MLPs represent a simple and widely used network architecture. In addition, individual layers are often used as part of more complex architectures, as practiced in this work.

2.2.2. Convolutional Layer

The convolutional neural network (CNN) is a network architecture developed substantially through the work of LeCun et al. [50], which has proven to be highly effective in image recognition. Convolutional layers consist of a receptive field and a kernel containing the weights. When processing image data, the receptive field slides over the input images with a given step size, generating many small sections multiplied by the kernel's weights. In contrast to fully connected layers, the filter uses the same weight matrix at different image locations. Thus, the number of parameters to be learned is drastically reduced. Moreover, convolutional layers focus on recognizing particularly relevant features and can detect them at different locations in an image [51]. In addition to processing 2D data such as images, convolutional layers can also be used to process 1D data such as time series or 3D data sets such as video sequences.

2.2.3. Recurrent Layer

Compared to fully connected layers, recurrent layers have feedback loops that allow the layer outputs to be fed back into the same layer again. This makes them highly suitable

for modeling sequential data such as text or time series. The longer the input sequences, the more feedback is required. In the case of very long sequences, this leads to deep networks. Training these networks often causes the problem where the gradient toward the lower layers shrinks and vanishes, or it grows in the other direction and explodes. These problems, also called vanishing and exploding gradients, cause the network to stop converging in the deeper layers or the training to become unstable [52]. Therefore, these networks are said to have a "short-term memory", which leads to the problem where longer-term dependencies are only barely considered or not at all.

To counter this problem, Hochreiter and Schmidhuber [53] developed the so-called long short-term memory (LSTM) cells. The special feature of LSTM cells compared to classical recurrent neurons is that they have an additional cell state. This cell state makes it possible to take long-term dependencies into account. The cell state is controlled by gates which decide what information is added to the state, what is forgotten, and how the cell state influences the network output.

2.2.4. Graph Neural Networks (GNNs)

The network layers described above are suitable for processing categorical data, sequential data, or data structured as rasters. However, many data sets are also available as networks in the form of graphs, in which the connection of individual objects to other objects plays an important role. These include, for example, social networks, molecules, traffic networks, or even sewer networks. A particular type of neural network was developed for this kind of data, the so-called graph neural network (GNN). Scarselli et al. [54] introduced this architecture over a decade ago. It was made especially popular by the work of Defferrard et al. [55] and Kipf and Welling [56], who combined GNNs with CNNs to form graph convolutional networks (GCNs). This architecture has been used to optimize the performance of neural networks for many problems including traffic forecasting [57], forecasting pressure in drinking water supply networks [58], and forecasting COVID-19 infection events [59].

The basis of GNNs is a graph G, which can be represented in the simplest form as G = (V; E), where V stands for the nodes and E for the edges. An edge from node $v_i \in V$ to node $v_j \in V$ can be described as $(v_i, v_j) \in E$. For efficient processing of graphs in ML applications, they are usually represented as a matrix. One way of doing this is to use an adjacency matrix $A \in \mathbb{R}^{N \times N}$ consisting of an $N \times N$ matrix in which, for each position i, j ($1 \leq i \leq N$; $1 \leq j \leq N$), the following is the case:

$$a_{i,j} = \begin{cases} 1 & \text{if} (v_i, v_j) \in E, \\ 0 & \text{else.} \end{cases} \quad (1)$$

Furthermore, graphs can be divided into directed and undirected. In directed graphs, the edges between two nodes can only be crossed in one direction, while in undirected graphs, the connection can be crossed in both directions. Furthermore, the edges in graphs can be weighted, whereby the entries in the adjacency matrix are multiplied by a given weight.

GNNs can be used to predict features on the level of nodes, edges, or the whole graph. In addition to the adjacency matrix, a feature matrix $X \in \mathbb{R}^{N \times D}$ is considered as input. Here, N describes the number of nodes and D is the number of input features per node. The output at the node $Z \in \mathbb{R}^{N \times F}$ (F stands here for the number of output features) is accordingly a function f of the adjacency and feature matrix:

$$Z = f(A; X) \quad (2)$$

The value of Z can be computed with different GNN architectures. Many studies rely on the GCNs described in Kipf and Welling [56]. GCNs transfer the convolution operation known from CNNs from image data to graph data. The main idea behind this is that the representation of a node always depends on its features and on the features of its neighboring nodes.

2.2.5. Generative Adversarial Networks

The generative adversarial network (GAN) is an architecture first presented by Goodfellow et al. [60] consisting of two sub-models. First, a generator G generates records based on a random distribution z similar to the training data set. Then, a classifier called discriminator D computes the probability of whether a data example x comes from the training data set or the generator. The models have contrary goals and compete against each other in a zero-sum game during training. The generator aims to produce outputs that the discriminator cannot distinguish as "real" contents from a data set or "fake" outputs produced by the generator. Conversely, the discriminator aims to classify the generator's outputs as "false" content with the highest possible probability. Both models are trained simultaneously and use the same loss function L, which indicates the likelihood of whether an input data set is "real" or "false." While the discriminator's parameters are adjusted to maximize this probability, the generator's parameters are adjusted to minimize it. This results in the following function presented in Goodfellow et al. [60]:

$$\min_{G}\max_{D} L(D,G) = E_{x\sim p_{data}(x)}[\log D(x)] + E_{z\sim p_z(z)}[\log(1 - D(G(z)))] \quad (3)$$

In classical GAN models, which receive only a random distribution as input, there is no way to control how data are generated [61]. With this in mind, Mirza and Osindero [61] developed conditional GANs (cGANs), a modified version, which in addition to a random distribution z, also depends on latent information y. This gives the model some context that can be used to influence the output. For example, images can be generated from contours or labels [22] or the resulting flooding from a precipitation forecast [34,37].

3. Case Study

Different deep-learning-based model setups were developed and tested for a study area to predict pluvial flooding. The data sets required for the training process were generated using an HD model of a study area and a data set with various precipitation events. In addition to the tests conducted to compare the model setups, this section also describes the HD model and the precipitation data set, as well as the data generation and preprocessing steps as an essential part of the research.

The deep learning models were developed in Python 3.8 using Tensorflow [62]. In addition, other common libraries such as Scikit-learn, Pandas, Numpy, and Matplotlib were used for data preprocessing and visualization. The two modules MIKE IO 1D and MIKE IO [63] were used to read the result files that are output by MIKE+. Geopandas and GDAL were also used to process spatial data and NetworkX for processing data structured as graphs. The models were trained on a workstation with an NVIDIA RTX 6000 GPU with 48 GB of GPU memory.

3.1. Study Area and Hydrodynamic Model

A study area of 3.1 km^2 in the south of the city of Gelsenkirchen in Germany was selected for the investigations (see Figure 2). The site is primarily urban and drains with a combined sewer system. The terrain has an average slope of 7.5% and is not influenced by rivers or slopes in terms of flooding. A railroad line runs across the area, dividing the catchment area into a northern and southern part. Both parts are connected by two underpasses, which are potentially at risk of flooding and were underwater during past extreme events.

To generate the flooding grids used as the target for the training process, a coupled 1D/2D HD model of the study area was implemented in the software Mike+ [64]. The municipal drainage utility of Gelsenkirchen provided a sewer network model for the study area. The field of the study area comprised 975 manholes and 982 reaches. A grid-based computational mesh with a 2 m × 2 m resolution was created, which contained 772,415 elements to model the runoff behavior at the ground surface. Elevation information was added to the computational mesh using 3D survey data acquired by airborne laser

scanning from the Cologne District Government [65]. Buildings were additionally raised so that they represented a flow barrier. The sewer network and surface models were then coupled in a bidirectional manner via the manholes.

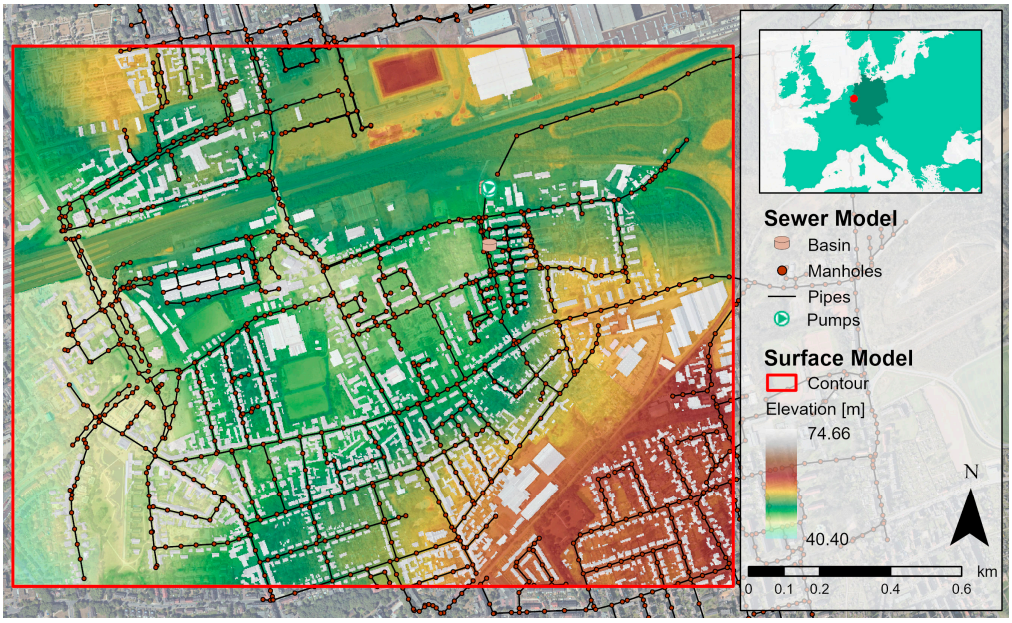

Figure 2. Illustration of the study area's sewer network and surface model (area outlined in red).

3.2. Pluvial Flood Event Data Sets

When using deep learning models, the developed model's accuracy depended heavily on the data set provided for the training process. For the training, precipitation hydrographs were used, for which the respective target variables were determined with the help of the hydrodynamic model. Since sewer networks in Germany were designed for overflow frequencies in the range of 2 to 10 years, according to DIN EN 752:2017 [66] and DWA-A 118 [67], precipitation events that lead to sewer system overflow are quite rare. For this reason, the investigations were not limited to historical events in the study area. Instead, data from a total of eight terrestrial rain gauges near the study area with continuously measured data for a period of >60 years as well as different design rainfall events were used. The considered rainfall data have a temporal resolution of five minutes.

To consider only relevant events, partial duration series were created from the rainfall records of the eight rain gauges. Preliminary experiments have shown that only rainfall events with a return period of >5 years are likely to cause overflow and the formation of relevant flood areas. Therefore, only rare events with higher return periods were considered. In total, 153 events suitable for training were identified.

While the real measured data provide a realistic representation of the rainfall characteristics, design rainfall data offer the possibility of a representative coverage of all relevant durations and return periods. Different model rainfall patterns, durations, and return periods were considered to cover the full range of all possible precipitation loads. Following Schmitt [68], so-called increase factors were also considered to cover precipitation beyond a return period of 100 years. The highest factor was set at 4.0 based on the findings from studies to determine "Maximized Area Precipitation Heights for Germany (MGN)". This is a physical–empirical-based estimate of the probable maximum physically possible precipitation heights [69].

As a result, 258 events (105 model rainfall events, 153 natural rainfall events) were available for model training. Figure 3 shows the distribution of events as a function of their respective return periods for the two data sets.

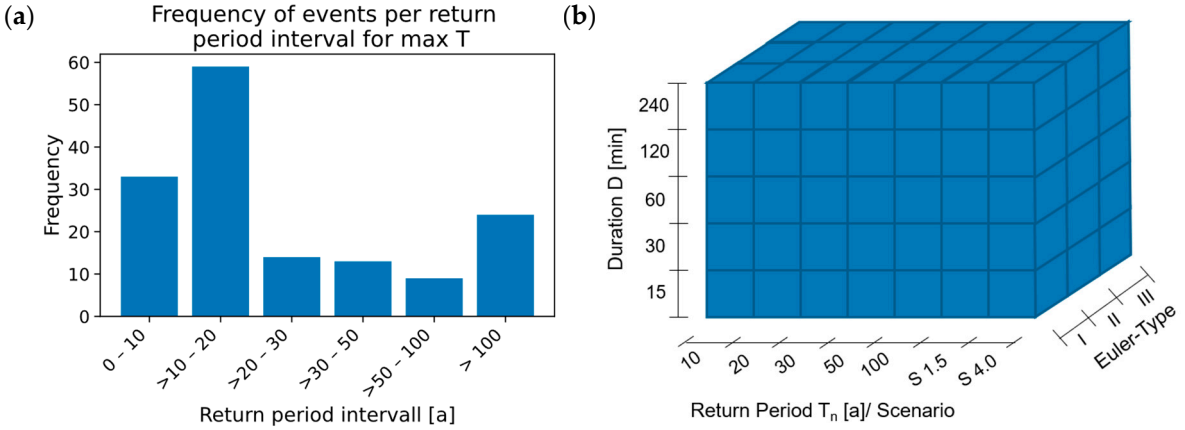

Figure 3. Distribution of events in the data set. (**a**) The maximum return times T distribution for all 153 natural rainfall events. (**b**) A schematic representation of the design rainfall events, with the selected durations, model rainfall types, and return periods/scenarios. For scenarios S 1.5 and S 4.0, the number indicates the increase factor by which the values of the 100-year model rains were multiplied.

3.3. Data Generation and Preprocessing

3.3.1. Data Generation Process

With the calibrated hydrodynamic 1D/2D flood model and the generated precipitation series as model load, the necessary training data sets for the ML-based forecast models were produced. The precipitation was assumed to be spatially homogeneous, and an additional lag time of 120 min was considered for each event to represent the decay of floods. As a result, different hydrographs such as overflow hydrographs from spilling manholes as well as sequences of grids with inundation areas were obtained.

At the end of a simulation, in addition to a map with the maximum water levels at the ground surface, MIKE+ outputs a multidimensional grid data set. The data set contained a temporal sequence of grids with the water levels at the respective time of the simulated event. This data set allowed training a ML method to predict the desired temporal evolution of an upcoming event and was used as a target variable in the training process. In addition, it was also possible to output overflow hydrographs from spilling manholes, which are considered as potential inputs in the analyses carried out here. For the models developed here, only overflow onto the ground surface was considered, not the inflow from surface runoff into the sewer network.

As in other studies [33,35], spatial information was used as potential input for the deep learning model in this study. Löwe et al. [35] conducted extensive investigations on the relevance of different types of spatial information in their study. The spatial information found to be most suitable (terrain aspect, curvature, the depth of terrain depressions, imperviousness, and flow accumulation) was also considered here in the analyses.

3.3.2. Data Preprocessing

Because of the different units and value ranges of the considered data and their partially right-skewed distributions, the data were further preprocessed. The spatial information was transformed following the procedure in Löwe et al. [35] and scaled to the interval $[-1, 1]$ if negative values were present, and to the interval $[0, 1]$ otherwise. The remaining data were also scaled to the interval $[0, 1]$, but no additional transformation was performed.

Predicting pluvial flooding was treated as a supervised learning problem in this study. Accordingly, the data for the training process were converted into pairs of input and target variables. The spatial information was relatively straightforward since it was static and did not change between training samples. However, this did not apply to the time series and grid sequences, which changed dynamically along the temporal dimension. Hence, a

sliding window approach was used. Thereby, for each time step t of an event, a window was opened over the past D time steps and the upcoming H time steps, resulting in intervals $[t_{-D+1},..., t]$ for the past time steps and $[t_{+1},..., t_H]$ for the predicted time steps. D and H were set to 60 min for the studies performed here, corresponding to 12 time steps for the chosen temporal resolution of five minutes. The precipitation forecast for the forecast horizon of 60 min was set to be the measured precipitation of the corresponding time steps for the investigations carried out here. In the future, a forecast generated by a precipitation forecast model will be used. The procedure for generating the training pairs P is shown as an example for one observation time step in Figure 4. The total number of all generated training pairs from the 258 used events was 9045.

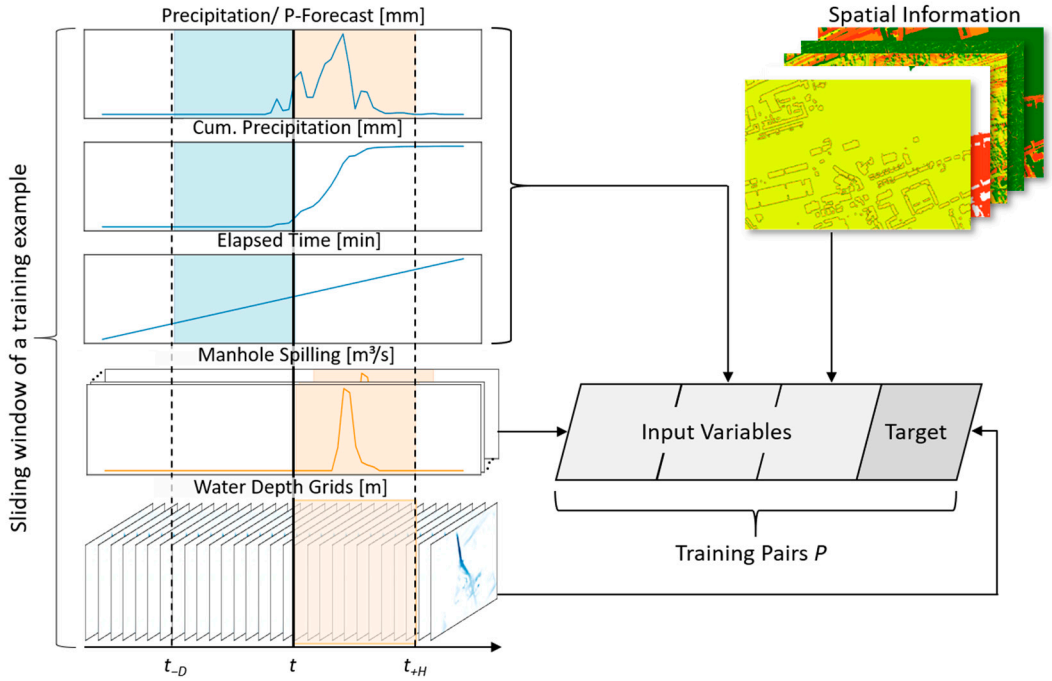

Figure 4. Converting the data into a supervised learning problem using a single training pair as an example.

The data set was split into training, validation, and testing data sets event by event. Out of the 258 events, the data pairs of 26 events were retained for testing, all of which were from the station closest to the study area. The data pairs from the remaining events were used 90% (209 events) for training and 10% (23 events) for validation.

3.4. Investigated Model Setups

As described in Section 2.1, artificial neural networks were used as ML methods to develop the prediction model. In addition, various potential inputs were available, which were examined to determine what extent the developed model would benefit from their integration. At the beginning of the investigations, the architecture shown in Figure 5 was chosen as a starting point. Initially, only precipitation information was selected as an input to predict a sequence of flooding grids. The architecture was inspired by the work of Guo et al. [33], but it underwent various modifications. In this model architecture, the precipitation information is first processed using two convolutional 1D layers for feature extraction before a fully connected layer and a reshaping layer follows. The latter converts the data into a format that can be upscaled to the output format. This is followed by a decoding part consisting of four deconvolutional 3D layers that generate the flooding raster sequence from the extracted features. Other architectures such as LSTM layers or fully connected layers for feature extraction or convolutional 3D layers in combination with upsampling layers for decoding were also tested, but they led to worse results. All

convolutional and deconvolutional layers, except the last one, are followed by a batch normalization layer [70] to stabilize the training process and to enable higher learning rates, as well as a rectified linear activation unit (ReLU) [71] as activation function. The last deconvolutional layer is followed by a sigmoid activation function [72] without batch normalization.

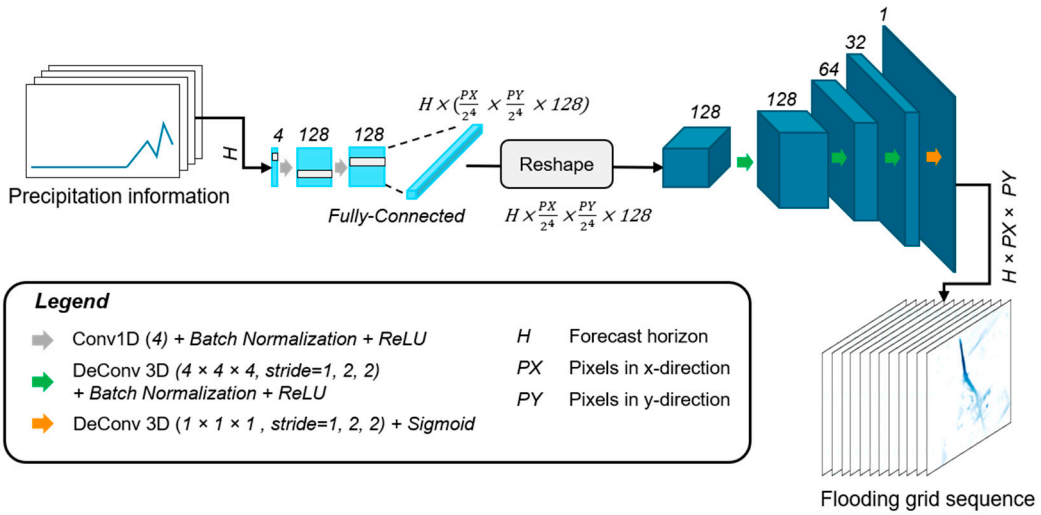

Figure 5. Baseline architecture for the following investigations.

For training, all models used the mean squared error as the objective function (unless otherwise described) and were trained with the Adam optimization algorithm [73] for 100 epochs. The size of the batches was set to 16 since larger batches led to GPU memory overload. A value of 0.001 was selected as the learning rate, previously determined following the procedure described by Smith [74]. Only the models with the smallest error for the validation data set during the training were saved to avoid overfitting.

3.4.1. Experiment 1: Comparison of Different Input Variables

In the first experiment, it was evaluated which combination of potential model inputs provided the best results. The precipitation information including the precipitation forecast was regarded as mandatory. The remaining two inputs were varied in all possible combinations so that the following models were compared with the following inputs:

- Model 1: Precipitation;
- Model 2: Precipitation + Overflow forecast;
- Model 3: Precipitation + Spatial information;
- Model 4: Precipitation + Overflow forecast + Spatial information.

Figure 6 shows the baseline architecture with the additional input paths considered for feature extraction. The overflow prediction is processed similar to the precipitation information and connected to the output architecture after the reshaping layer via a concatenate layer before the decoding path follows. At the same point, the spatial information is integrated into the network. The feature extraction for this type of data is conducted with an encoder structure consisting of several convolutional 2D layers with a stride of two to downsample the input raster. The output of the last convolutional 2D layer is stacked H times to obtain identical dimensions in front of the concatenate layer.

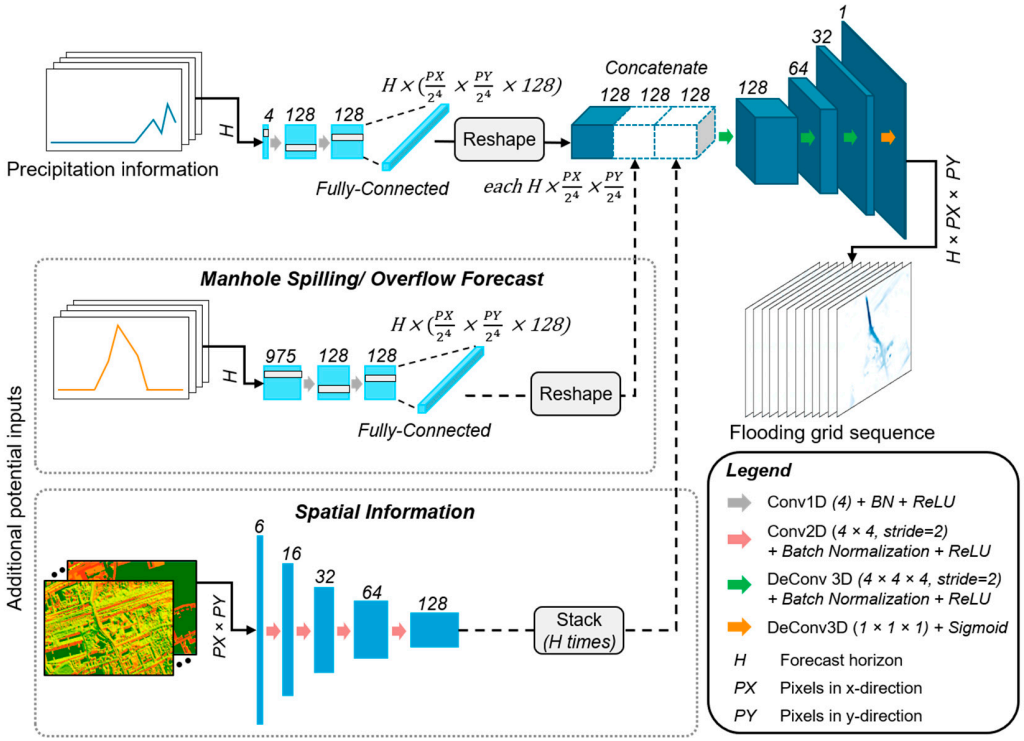

Figure 6. Baseline architecture combined with the corresponding input paths for the overflow forecast and the spatial information.

3.4.2. Experiment 2: Comparison of Different Preprocessing of the Overflow Data

Different formats to integrate the overflow data into the model were investigated in the second experiment. Initially, these were available as hydrographs for all nodes in the catchment area. The issue was to what extent the model could benefit from these several hundred hydrographs without spatial relations. In this context, in addition to the unstructured overflow hydrographs (variant a), two other variants were investigated, adding the overflow data to the model as a raster sequence (variant b) and as a spatiotemporal graph (variant c). Figure 7 provides an overview of the possible architectures.

The raster sequences in variant (b) were created by intersecting the overflow hydrographs with a sink catchment raster. The result was a sequence of grids with the accumulated overflow volumes of all manholes per time step and for each sink catchment. In the model architecture, the raster sequences are processed with an encoder structure consisting of convolutional 3D layers with a stride of two. As in the decoding part, each layer is followed by a batch normalization layer and a ReLU activation function. The output of the last layer is then concatenated to the baseline architecture and further processed there.

In variant (c), the overflow forecast is processed with a temporal graph convolutional network (T-GCN) following Zhao et al. [75] and Yu et al. [57]. This approach combines a graph convolutional layer with a recurrent layer. While the graph convolutional layer captures the spatial dependencies of the sewer network, the recurrent layer captures the temporal dynamics of the overflow process at the individual manholes. This enables the modeling of the spatiotemporal learning problem presented here. The graph convolutional layer receives a feature matrix containing the overflow hydrographs and an adjacency matrix representing the sewer network as an unweighted and directed graph. A LSTM layer is used as the recurrent layer. The output of the T-GCN block is then passed to a fully connected layer followed by a reshaping layer, analogous to the precipitation data, before concatenating with the output architecture.

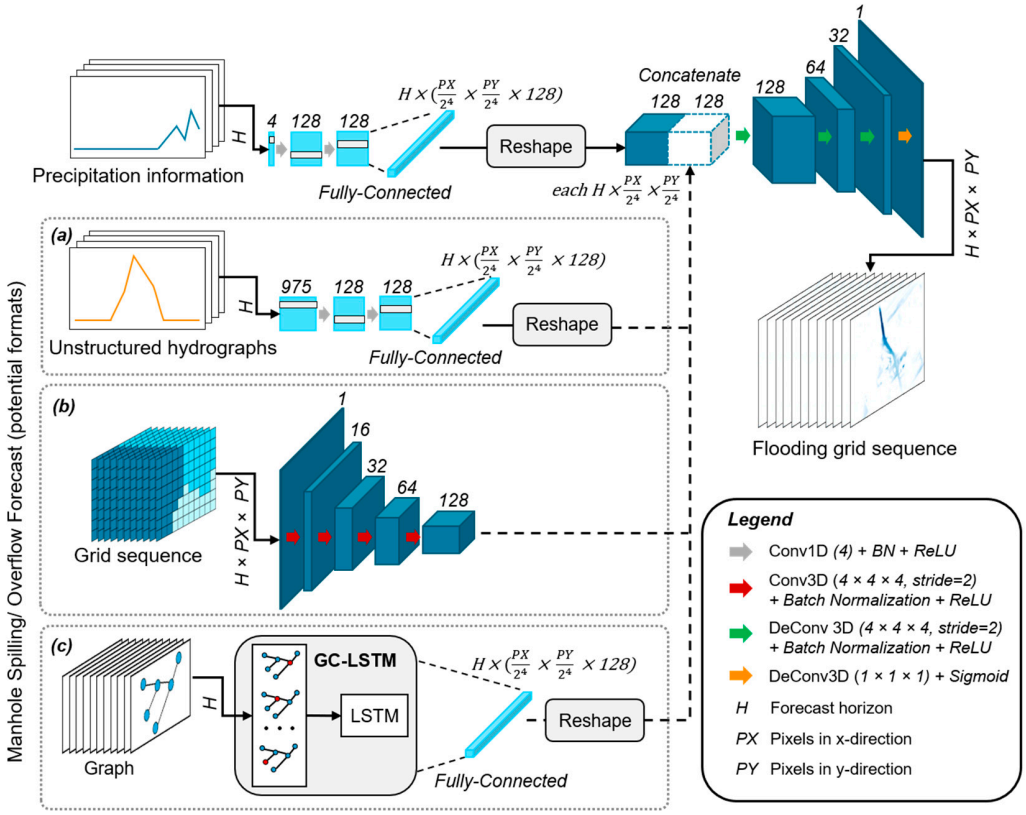

Figure 7. Baseline architecture with the different input paths for the considered formats of the overflow forecast. Baseline architecture with the input paths for the overflow forecast formatted as (**a**) unstructured hydrographs, (**b**) raster sequences, and (**c**) spatiotemporal graphs.

3.4.3. Experiment 3: Comparison of Different Model Setups

In a third experiment, the performance of the previously best-evaluated model was compared to a conditional generative adversarial network (cGAN). The structure of the cGAN is shown in Figure 8 and was inspired by the work of Isola et al. [22] and Hofmann and Schüttrumpf [34]. The latter had used the architecture successfully for flood prediction. Unlike a normal GAN, the cGAN receives context in addition to noise as input. In the present investigations, following the findings of Isola et al. (2017), the noise was completely ignored as input and only context in the form of the potential model inputs from experiment 1 was considered. Moreover, following similar studies [22,34,37], a mean absolute error function (L1 loss) was integrated into the objective function (cf. Formula (3)). Thus, the generator aims not only to fool the discriminator, but also to minimize the error between the results of the HD-Model used as the target variable.

The best model from experiments 1 and 2 was used as the architecture for the generator. A network structure suitable for classification was used as the discriminator. It first extracts the features from the individual inputs similar to the other model structures and then merges them with a concatenate layer (see Figure 9). Afterward, another convolutional 3D block with ReLU and batch normalization follows, as well as a convolutional 3D layer followed by a sigmoid activation function. The output represents a binary classification. In contrast to the generator, dropout [76] with a dropout rate of 0.5 was used for regularization in the discriminator. In this way, the training process could be stabilized and better results could be achieved.

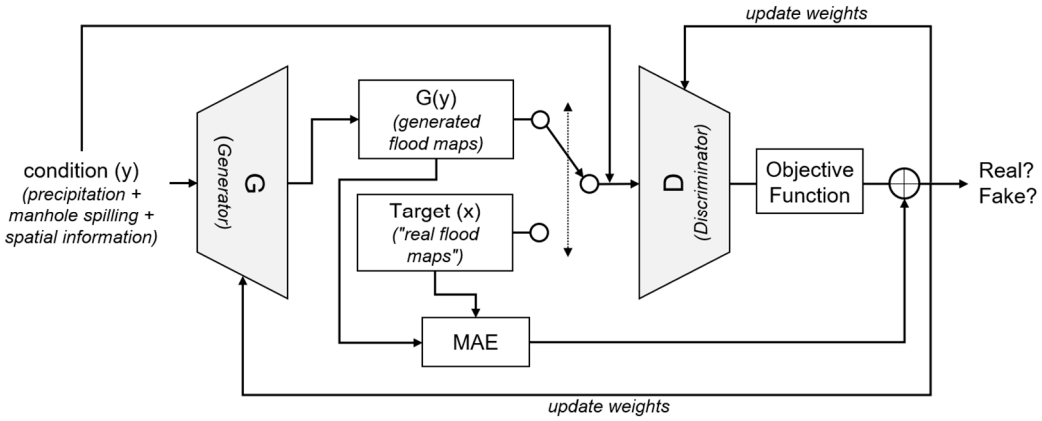

Figure 8. Architecture of the conditional GAN.

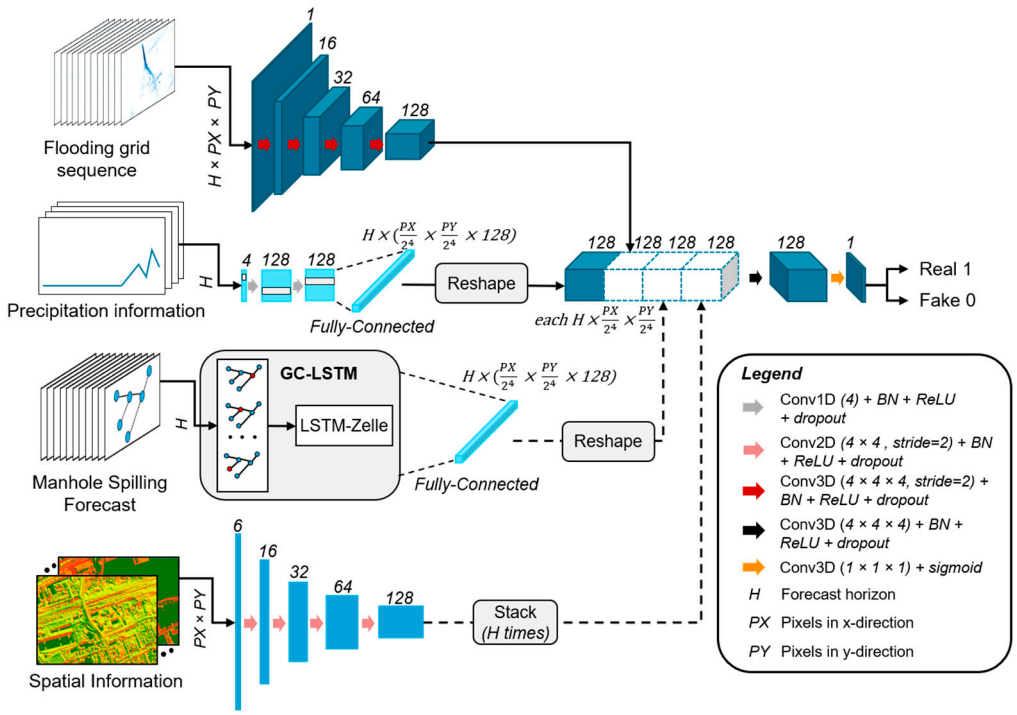

Figure 9. Architecture of the discriminator.

3.5. Performance Evaluation

The predicted (ML model) and simulated (HD model) flooding grids were compared cell by cell to evaluate the prediction results. For this purpose, the root mean squared error (RMSE) and the critical success index (CSI) were used as two different quality criteria types.

The RMSE is a continuous index that compares the exact water levels and evaluates the average deviation:

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i^{NN} - y_i^{HD}\right)^2} \quad (4)$$

where n stands for the number of cells compared and y_i for the respective values of the individual cells determined with the neural network NN and the hydrodynamic model HD. The RMSE can assume values in the range [0, ∞], where 0 corresponds to the optimal fit. The absolute error is given as the result. Other metrics for determining the relative error, such as the relative mean squared error (MRSE), were also tested. However, it was found that pixels with low water levels sometimes resulted in extreme relative errors. In the subsequent averaging of error values over all the cells of a flooding grid, this problem

led to poor results. However, the affected cells have only a low hazard potential and are thus of minor relevance compared to cells with high water levels.

The CSI is a categorical index for evaluating location accuracy and is a widely used measure for assessing extreme events in both precipitation [43–45] and flash flood forecasting [35,77]. Compared to other categorical indices such as the hit rate or the false alarm rate the CSI considers both misses and false alarms. Since both are equally unfavorable for the developed prediction model, the CSI is best suited for this purpose. First, binary classification of the cells needs to be performed to determine the CSI. In the present case, the pixels were classified as flooded and non-flooded. Subsequently, the CSI was calculated as follows:

$$\mathrm{CSI} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN} + \mathrm{FP}} \tag{5}$$

In this example, TP stands for the number of cells correctly predicted to be flooded, FP denotes the cells incorrectly predicted to be flooded, and FN indicates the number of cells incorrectly predicted not to be flooded. It thus responds to both missed and false alarms. This makes it well suited for the present task since missed and falsely predicted inundated areas are equally inconvenient in an emergency. The values of the CSI are in the interval [0, 1], where 1 corresponds to the best result.

The evaluation procedure only calculated metrics for pixels where the HD model or the neural network predicted water levels above a given threshold. Following the procedure in Löwe et al. [35] or as common practice in precipitation forecasting [43], multiple thresholds were considered for the CSI to evaluate the location accuracy at different water levels. The same approach was used here for the RMSE to account for the deviation in dependence on various water levels.

4. Results

4.1. Comparison of the Investigated Model Setups

The three experiments described were carried out one after another, and in each case, the best model was carried into the next experiment. Table 1 summarizes the results of all experiments. The metrics for the individual water level threshold values d were formed in each case as the mean value of all samples of the 26 test events and all prediction time steps. For experiment 1, it was clearly shown that considering the overflow forecast (models 2 and 4) led to better results and generally predicted high water levels more reliably. The additional consideration of spatial information led to higher accuracy for water depth values $\leq$ 0.2 m. On the other hand, at higher thresholds, model 2 without spatial information performed best. Altogether, the difference between the two models was small. The result was unsurprising because the spatial information was a static variable without changes among individual training pairs. Thus, the data set acted more as a mask and did not significantly impact the error signal required for training progress. Nevertheless, since various studies have shown that spatial information can enable the transferability of trained models [35,78], model 4 was included in the following experiment.

A comparison of the different model setups in experiment 2 showed that considering the overflow information as a raster sequence led to the worst results. In addition, the format demanded a significantly larger memory consumption during model training and almost double the computation time. The unstructured input of the overflow hydrographs performed slightly better for the lower thresholds, although the differences were marginal, especially for the RMSE. The input as a graph gave the best results for the higher thresholds, which was the most relevant for flood prediction. Therefore, model 7 was carried into the final experiment.

Table 1. Evaluation results for all models from the three experiments (for each experiment and metric, the best result is bolded).

Model	RMSE ↓					CSI ↑				
	d ≥ 0.02	**d ≥ 0.05**	**d ≥ 0.1**	**d ≥ 0.2**	**d ≥ 0.5**	**d ≥ 0.02**	**d ≥ 0.05**	**d ≥ 0.1**	**d ≥ 0.2**	**d ≥ 0.5**
Experiment 1: Model Inputs										
Model 1 (Inputs: rain)	0.039	0.052	0.074	0.140	0.553	0.504	0.488	0.399	0.299	0.122
Model 2 (Inputs: rain, manhole spilling)	0.028	0.037	0.052	0.096	**0.096**	0.538	0.543	0.495	0.414	**0.768**
Model 3 (Inputs: rain, spatial information)	0.037	0.050	0.074	0.144	0.547	0.510	0.466	0.384	0.293	0.157
Model 4 (Inputs: rain, spatial information, manhole spilling)	**0.026**	**0.035**	**0.051**	**0.092**	0.118	**0.595**	**0.574**	**0.511**	**0.421**	0.746
Experiment 2: Manhole Spilling Forecast Format										
Model 5 (Unordered)	**0.026**	**0.035**	**0.051**	0.094	0.118	**0.595**	**0.574**	**0.511**	0.421	0.746
Model 6 (Raster Sequence)	0.030	0.040	0.058	0.115	0.148	0.548	0.514	0.434	0.340	0.679
Model 7 (Graph)	0.026	0.036	0.052	**0.092**	**0.081**	0.575	0.557	0.492	**0.424**	**0.788**
Experiment 3: Model Architecture										
Model 8 (T-GCN)	**0.026**	**0.036**	**0.052**	**0.092**	**0.081**	0.575	0.557	0.492	0.424	**0.788**
Model 9 (T-GCN cGAN)	0.027	0.037	0.055	0.113	0.158	**0.623**	**0.602**	**0.545**	**0.440**	0.723

The third and last experiment showed high location accuracy for the model setup as a conditional GAN (model 9). On the other hand, the "classic" T-GCN (model 8) showed a higher accuracy for the RMSE. For a better assessment of individual outliers that may negatively affect the metrics, further evaluations were subsequently performed with both models.

4.2. Assessment of the Prediction Accuracy

The metrics were determined for each event in a further step to obtain a more detailed evaluation of the model performance. Figure 10 shows the distribution of the results for the T-GCN and the T-GCN cGAN. It should be pointed out that only 4 of the 26 events resulted in water levels > 0.5 m, so the metrics determined for that threshold are only partially representative. The differences between the two models shown in Table 1 are also apparent. Moreover, a slightly more extensive spreading of metrics was observed at higher thresholds for the T-GCN cGAN than for the T-GCN. Still, a significant negative influence of extreme outliers on the results could not be detected.

For a threshold value of 0.05 m, Figure 11 shows the RMSE and the CSI values of the 26 events with respect to their return periods (T). Here, both models provided good results even for the particularly relevant very rare events with T > 100 a. For the CSI, the results were in the upper range of all test events. The results for the RMSE were also positive considering that the absolute deviations were included in the calculation. Neither of the two models presented extreme outliers at certain recurrence intervals, and their results were similar.

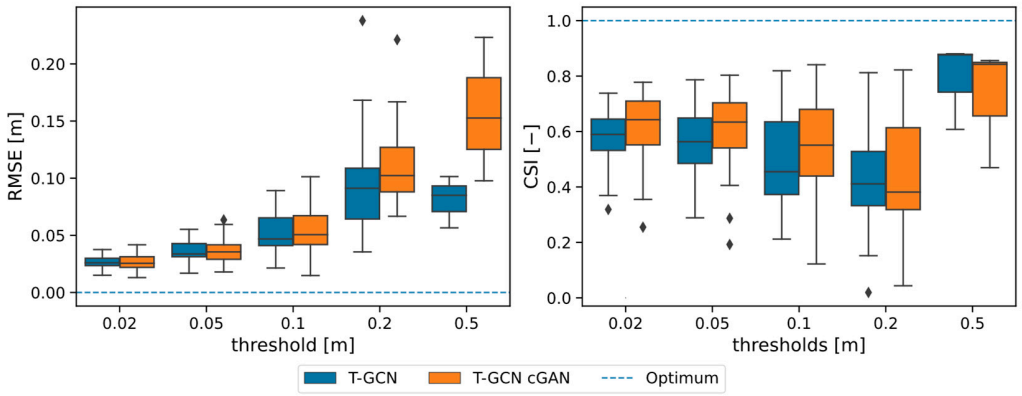

Figure 10. Distribution of metrics over all 26 events in the test data set.

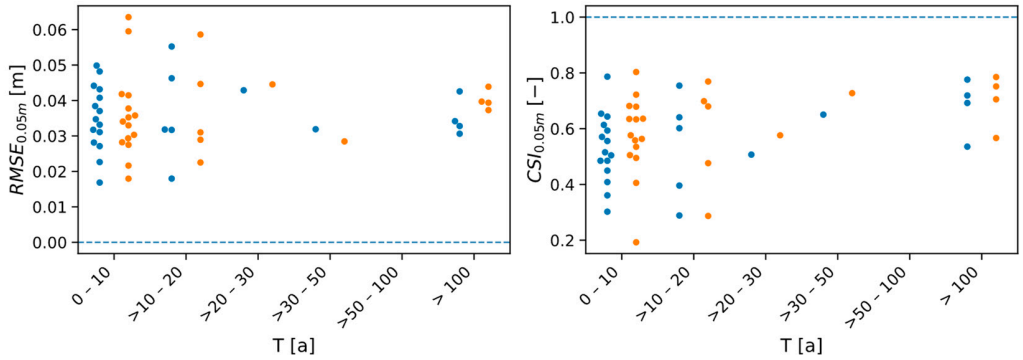

Figure 11. Distribution of metrics depending on different recurrence intervals for the threshold value d $\geq$ 0.5 m.

In a third analysis, it was assessed whether the models overestimated or underestimated water levels. For this purpose, the prediction error as a function of the simulated water levels is shown in Figure 12 in a 2D histogram. All pixels from all forecast grids of the 26 events above a threshold of 0.05 m were considered. The dashed line indicates the ideal fit between forecast and HD simulation. The deviations vary relatively evenly around the dashed line for both models, with a slight trend toward overestimating water levels. Again, this showed a slightly better performance of the T-GCN.

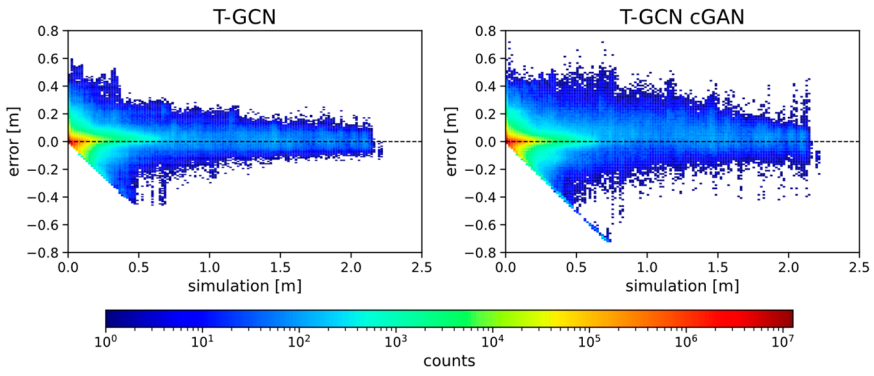

Figure 12. The 2D histogram with the prediction error as a function of the simulated water depths.

4.3. Forecast for a Historical Heavy Rainfall Event

For the final evaluation, the T-GCN was tested using the historical heavy rainfall event of 3 July 2009 in the study area of Gelsenkirchen (Figure 13).

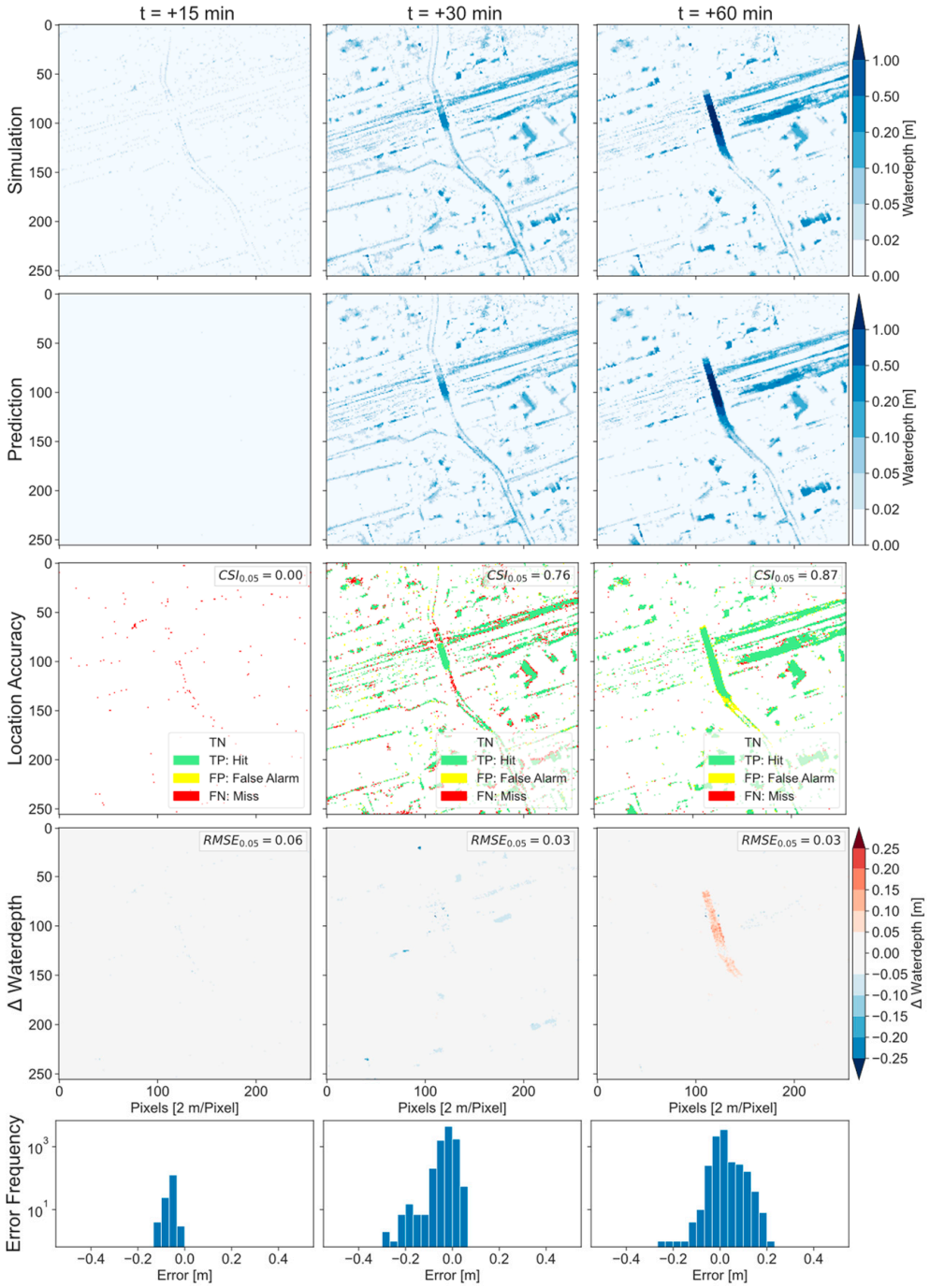

Figure 13. Results and evaluation for three time steps of a single forecast with the T-GCN at the beginning of the event on 3 July 2009 in Gelsenkirchen. Instead of showing the entire study area, a section with a flooded underpass is expanded for better visualization.

A forecast was generated at the beginning of the event for a forecast horizon of 60 min. The precipitation sum for the predicted period was 46 mm and for some duration intervals, return periods of more than 200 years were reached. Figure 13 presents HD simulation and forecast results, location accuracy and the water depth difference for three forecast time steps. The visual comparison shows a high agreement and that no unrealistic flooding patterns were produced. In addition, the following characteristics of the model are shown in the figure:

1. Predictions with shallow water depths and only a few flooded pixels often lead to large errors. This problem is particularly evident at step t = +15 min with a CSI of 0, the worst possible result. The RMSE also shows the worst value compared to the other time steps. The same problem was also found by Löwe et al. [35]. On the other hand, predicted flood maps with many flooded pixels usually show a high accuracy, as is the case for time steps t = +30 min and + 60 min. Accordingly, the flooding patterns particularly relevant for crisis management are predicted with high accuracy.
2. The model reacts with a slight delay to the precipitation load. While increasing flood areas before the peak are underestimated, the extent of areas after the peak is slightly overestimated. This behavior is illustrated by the histogram with the error frequencies, and it is also displayed in other events.
3. In the center of the depicted section, there is an underpass where the most considerable differences of up to 25 cm occur. However, it should be noted that the water levels there are sometimes more than two meters high. In this case, the relative error would be in the range of about 10–15% and thus within an acceptable range.

5. Discussion

The results showed a good agreement between the predicted inundation areas and the HD model results. The visual comparison showed only small differences, which should play only a minor role in its use in warning systems or as a basis for decisions in crisis management. The RMSE and CSI values were in a range similar to other studies [34,35,37], where deep learning models were used to compute flood maps. However, it should be noted that the comparison with other studies is limited because of the different prediction model tasks (grid sequences instead of single grids) and the different characteristics of the used study areas (topography, imperviousness, etc.). The investigations also found that the model provided good results for rare events with high return periods. In addition, the chosen model structure is independent of the event duration. That is because the model does not require inputs such as the precipitation load for the entire event, but only considers a fixed number of past and future time steps of each forecast time point in an event. On the other hand, it must be considered that perfect precipitation and overflow forecasts were used in the experiments. In practice, both predictions are subject to uncertainties, which would affect the model developed here. Therefore, regarding operational use, further investigations with results from forecast models for precipitation and overflow are required to evaluate the effects of the uncertainties on the predicted inundation areas.

The final model needed only a few seconds to calculate the flooding sequence for the following 60 min. Even when extended to larger areas, the computation time is expected to be less than one minute, making the model suitable for real-time operation. This result was consistent with findings from other studies [31,34,35,78]. Although an extension of the prediction horizon will lead to higher computational and memory requirements because of the additional flooding grids, it will be otherwise technically feasible. However, the possible forecast horizon is limited by the uncertainties of precipitation forecasts. These increase dramatically after only a few minutes for extreme events using the currently available forecast models and thus do not allow meaningful forecasts for more than two hours [79].

As Hofmann and Schüttrumpf [34] indicated, generation of the training, validation, and test data sets with hydrodynamic models is very time intensive. For the relatively small test area used in this study, it took about two months to compute all 258 considered events on a high-performance workstation. If extended to the entire city area of Gelsenkirchen, without additional computing resources, the calculations would need several years, making the approach unfeasible. In addition to more extensive computing resources, a smaller data set could be another solution. For this purpose, investigations are planned on determining the data quantity required to achieve adequate prediction results. Especially for the events with lower return times, it is quite possible that reducing the data set will not lead to a significant loss of quality.

In addition to the required computational resources for data generation, the scalability of the model was limited by the high memory requirements for model training. Because of the high dimensionality of the flooding sequences used as target size (12 images of 1024 × 768 px for the relatively small study area of 3.1 km^2), an expansion to larger areas and thus a higher number of raster cells to be computed will quickly exceed the available GPU RAM. Considering the currently available technology, this makes training impossible beyond a certain amount of data. One approach to covering entire urban areas is to divide the computational domain into sub-models and merge the results as described in Berkhahn et al. [31]. Another method is to train models in parallel on multiple GPUs. One approach to this is represented by the Python library Mesh-TensorFlow [80], which allows developing large models with extreme memory requirements and training them in parallel on multiple GPU units.

Another way to scale the model to an entire urban area is to make a trained model transferable. A model could then be developed for a sub-area and used to forecast the rest of the municipal area. Furthermore, this characteristic would allow the model to adapt to changes in the catchment area, for example, in topography, land use, or the sewer network, without retraining. Some studies [33,35] used topographic information as an additional input to include physical system properties to establish model transferability. This approach can be combined with transfer learning techniques to improve results for the target area with a small amount of additional training [36]. Because of the further consideration of overflow as an input variable in the experiments presented here, considering only topographic information is insufficient. Instead, the combination or intersection of the overflow forecast with other physical system properties is required as an additional input to enable transferability. The intersection of overflow forecasts with sink catchments performed in experiment 2 (see Section 3.4.2) is a step in this direction. A similar approach was proposed by Löwe et al. [35] by weighting flow paths by adjacent overflow volumes in a raster data set. However, both methods are likely to yield losses in prediction accuracy and lead to a significant increase in GPU memory demand.

Finally, with the current model setup, the neural network can only become as good as the HD model. Accordingly, the generally known limitations of HD models also apply here. These include the limited validation opportunity due to the lack of measurement networks for recording water levels during flooding events. Water level detection using social media images [4,5] or recordings from surveillance cameras [6] could provide a solution in the future. In addition, spatial information on flooding extent would be highly desirable such as through extraction from satellite data [81,82]. Nevertheless, because of its coarse temporal and spatial resolution, this approach is currently limited to fluvial or tidal flooding. In the future, further technical developments might enable the use of satellite data for pluvial flood modeling and possibly even provide a data source for training ML models.

6. Conclusions

This paper presented experiments with different deep learning models for predicting pluvial flooding in urban areas. The special feature of the different models was the spatial and temporal prediction of the flooding situation in the form of a sequence of flood maps. In addition, the models generated a forecast for the following 60 min at any given time step of an event. As part of the model development, experiments were conducted to determine the influence of different input variables, input formats, and model architectures on prediction quality. The best model proved to be a T-GCN, which was characterized by low computation times and, at the same time, produced flood maps with reasonable differences from the ones produced by an HD model. Furthermore, the best results were achieved with precipitation information, an overflow forecast, and spatial information as input. The main disadvantage of the presented model setup is the limited scalability and transferability. On the one hand, long computation times during training data generation and high demand for GPU RAM during model training limit the size of the considered area. On the other hand, due to the limited transferability, the trained model cannot be

used for other catchments and must be retrained when structural changes occur in the used catchment area. The additional consideration of overflow, which positively affects the model quality, means that the trained model cannot be transferred without further input information. Here, additional information about the spatial overflow structure is required as model input, which can be exchanged if the model is used for other catchment areas.

Further investigations will be needed at various points. This applies mainly to integrating real rainfall and overflow forecasts into the prediction process. Here, for use in real-time warning systems, it must first be shown that accurate flood forecast results can still be achieved despite the uncertainties transferred to the model. The likely lower model quality can be mitigated by systematic hyperparameter tuning, which was not part of this study. To the best of our knowledge, no other studies currently use machine learning models to predict flooding sequences considering the sewer network. Accordingly, no models were available that could be used as a benchmark in this study. Therefore, a transfer to an open-source dataset such as the Belinge dataset [83] is envisaged to provide the T-GCN as a benchmark for future developments.

Furthermore, the model's scalability to an entire urban area must be examined. The time required for data generation by the HD model and the high GPU memory requirements for training remain constraints, but they can be countered with various approaches such as increased computing resources or targeted data volume reduction. It also remains to be examined how and to what extent it is possible to scale the model to large areas while considering manhole spilling as an input variable. In addition, further investigations are needed regarding the accuracy of the developed model in areas with different topography.

Author Contributions: Conceptualization, B.B. and M.Q.; methodology, B.B., J.H. and J.K.d.S.; software, B.B.; formal analysis, B.B.; investigation, B.B.; writing—original draft preparation, B.B.; writing—review and editing, J.H., J.K.d.S., A.N. and M.Q.; visualization, B.B.; supervision, J.H., A.N. and M.Q.; project administration, M.Q.; funding acquisition, M.Q. All authors have read and agreed to the published version of the manuscript.

Funding: This research work is part of the research project "KIWaSuS—KI-basiertes Warnsystem vor Starkregen und urbanen Sturzfluten" (AI-based warning system for heavy rain and urban flash floods) funded by the Federal Ministry of Education and Research (BMBF), Germany (grant number 13N15556).

Data Availability Statement: Restrictions apply to the availability of the used data. Data from various institutions were used for the research, and the authors committed to not passing them on to third parties.

Acknowledgments: The authors thank Emschergenossenschaft and Abwassergesellschaft Gelsenkirchen mbH for providing the used precipitation data and the hydrodynamic sewer model.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Seneviratne, S.I.; Zhang, X.; Adnan, M.; Badi, W.; Dereczynski, C.; Di Luca, A.; Ghosh, S.; Iskandar, I.; Kossin, J.; Lewis, S.; et al. Weather and Climate Extreme Events in a Changing Climate. In *Climate Change 2021: The Physical Science Basis. Contribution of Working Group I to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change*; Masson-Delmotte, V., Zhai, P., Pirani, A., Connors, S.L., Péan, C., Berger, S., Caud, N., Chen, Y., Goldfarb, L., Gomis, M.I., et al., Eds.; Cambridge University Press: Cambridge, UK; New York, NY, USA, 2021; pp. 1513–1766.
2. Rosenzweig, B.R.; McPhillips, L.; Chang, H.; Cheng, C.; Welty, C.; Matsler, M.; Iwaniec, D.; Davidson, C.I. Pluvial flood risk and opportunities for resilience. *WIREs Water* **2018**, *5*, e1302. [CrossRef]
3. Zhao, G.; Balstrøm, T.; Mark, O.; Jensen, M.B. Multi-Scale Target-Specified Sub-Model Approach for Fast Large-Scale High-Resolution 2D Urban Flood Modelling. *Water* **2021**, *13*, 259. [CrossRef]
4. Wang, Y.; Chen, A.S.; Fu, G.; Djordjević, S.; Zhang, C.; Savić, D.A. An integrated framework for high-resolution urban flood modelling considering multiple information sources and urban features. *Environ. Model. Softw.* **2018**, *107*, 85–95. [CrossRef]
5. Chaudhary, P.; D'Aronco, S.; Moy de Vitry, M.; Leitão, J.P.; Wegner, J.D. Flood-Water level estimation from social media images. *ISPRS Ann. Photogramm. Remote Sens. Spatial Inf. Sci.* **2019**, *IV-2/W5*, 5–12. [CrossRef]
6. Moy de Vitry, M.; Kramer, S.; Wegner, J.D.; Leitão, J.P. Scalable flood level trend monitoring with surveillance cameras using a deep convolutional neural network. *Hydrol. Earth Syst. Sci.* **2019**, *23*, 4621–4634. [CrossRef]

7. Zischg, A.P.; Mosimann, M.; Bernet, D.B.; Röthlisberger, V. Validation of 2D flood models with insurance claims. *J. Hydrol.* **2018**, *557*, 350–361. [CrossRef]
8. Henonin, J.; Russo, B.; Mark, O.; Gourbesville, P. Real-time urban flood forecasting and modelling—A state of the art. *J. Hydroinf.* **2013**, *15*, 717–736. [CrossRef]
9. Giannaros, C.; Dafis, S.; Stefanidis, S.; Giannaros, T.M.; Koletsis, I.; Oikonomou, C. Hydrometeorological analysis of a flash flood event in an ungauged Mediterranean watershed under an operational forecasting and monitoring context. *Meteorol. Appl.* **2022**, *29*, e2079. [CrossRef]
10. Faure, D.; Schmitt, P.; Auchet, P. Limits of radar rainfall forecasting for sewage system management: Results and application in Nancy. In Proceedings of the 8th International Conference on Urban Storm Drainage, Sydney, Australia, 30 August 1999.
11. Quirmbach, M. Nutzung von Wetterradardaten für Niederschlags- und Abflussvorhersagen in Urbanen Einzugsgebieten. Ph.D. Thesis, Ruhr-Universität Bochum, Bochum, Germany, 2003.
12. Jasper-Tönnies, A.; Hellmers, S.; Einfalt, T.; Strehz, A.; Fröhle, P. Ensembles of radar nowcasts and COSMO-DE-EPS for urban flood management. *Water Sci. Technol.* **2017**, *2017*, 27–35. [CrossRef]
13. René, J.-R.; Djordjević, S.; Butler, D.; Mark, O.; Henonin, J.; Eisum, N.; Madsen, H. A real-time pluvial flood forecasting system for Castries, St. Lucia. *J. Flood Risk Manag.* **2015**, *11*, 269–283. [CrossRef]
14. Hofmann, J.; Schüttrumpf, H. Risk-Based and Hydrodynamic Pluvial Flood Forecasts in Real Time. *Water* **2020**, *12*, 1895. [CrossRef]
15. Neal, J.C.; Fewtrell, T.J.; Bates, P.D.; Wright, N.G. A comparison of three parallelisation methods for 2D flood inundation models. *Environ. Model. Softw.* **2010**, *25*, 398–411. [CrossRef]
16. Leandro, J.; Chen, A.S.; Schumann, A. A 2D parallel diffusive wave model for floodplain inundation with variable time step (P-DWave). *J. Hydrol.* **2014**, *517*, 250–259. [CrossRef]
17. Bates, P.D.; Horritt, M.S.; Fewtrell, T.J. A simple inertial formulation of the shallow water equations for efficient two-dimensional flood inundation modelling. *J. Hydrol.* **2010**, *387*, 33–45. [CrossRef]
18. Jamali, B.; Löwe, R.; Bach, P.M.; Urich, C.; Arnbjerg-Nielsen, K.; Deletic, A. A rapid urban flood inundation and damage assessment model. *J. Hydrol.* **2018**, *564*, 1085–1098. [CrossRef]
19. Xu, K.; Ba, J.; Kiros, R.; Cho, K.; Courville, A.; Salakhutdinov, R.; Zemel, R.; Bengio, Y. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. *arXiv* **2015**, arXiv:1502.03044.
20. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. *arXiv* **2015**, arXiv:1505.04597.
21. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778, ISBN 9781467388511.
22. Isola, P.; Zhu, J.-Y.; Zhou, T.; Efros, A.A. Image-to-Image Translation with Conditional Adversarial Networks. In Proceedings of the 30th IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–27 July 2017.
23. Reichstein, M.; Camps-Valls, G.; Stevens, B.; Jung, M.; Denzler, J.; Carvalhais, N.; Prabhat. Deep learning and process understanding for data-driven Earth system science. *Nature* **2019**, *566*, 195–204. [CrossRef]
24. Shen, C. A Transdisciplinary Review of Deep Learning Research and Its Relevance for Water Resources Scientists. *Water Resour. Res.* **2018**, *54*, 8558–8593. [CrossRef]
25. Mosavi, A.; Ozturk, P.; Chau, K. Flood Prediction Using Machine Learning Models: Literature Review. *Water* **2018**, *10*, 1536. [CrossRef]
26. Bentivoglio, R.; Isufi, E.; Jonkman, S.N.; Taormina, R. Deep Learning Methods for Flood Mapping: A Review of Existing Applications and Future Research Directions. *Hydrol. Earth Syst. Sci.* **2022**, *26*, 4345–4378. [CrossRef]
27. Bermúdez, M.; Ntegeka, V.; Wolfs, V.; Willems, P. Development and Comparison of Two Fast Surrogate Models for Urban Pluvial Flood Simulations. *Water Resour. Manag.* **2018**, *32*, 2801–2815. [CrossRef]
28. Jhong, B.-C.; Wang, J.-H.; Lin, G.-F. An integrated two-stage support vector machine approach to forecast inundation maps during typhoons. *J. Hydrol.* **2017**, *547*, 236–252. [CrossRef]
29. Lin, G.-F.; Lin, H.-Y.; Chou, Y.-C. Development of a real-time regional-inundation forecasting model for the inundation warning system. *J. Hydroinf.* **2013**, *15*, 1391–1407. [CrossRef]
30. Bermúdez, M.; Cea, L.; Puertas, J. A rapid flood inundation model for hazard mapping based on least squares support vector machine regression. *J. Flood Risk Manag.* **2019**, *12*, e12522. [CrossRef]
31. Berkhahn, S.; Fuchs, L.; Neuweiler, I. An ensemble neural network model for real-time prediction of urban floods. *J. Hydrol.* **2019**, *575*, 743–754. [CrossRef]
32. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef]
33. Guo, Z.; Leitão, J.P.; Simões, N.E.; Moosavi, V. Data-driven flood emulation: Speeding up urban flood predictions by deep convolutional neural networks. *J. Flood Risk Manag.* **2020**, *14*, e12684. [CrossRef]
34. Hofmann, J.; Schüttrumpf, H. floodGAN: Using Deep Adversarial Learning to Predict Pluvial Flooding in Real Time. *Water* **2021**, *13*, 2255. [CrossRef]
35. Löwe, R.; Böhm, J.; Jensen, D.G.; Leandro, J.; Rasmussen, S.H. U-FLOOD—Topographic deep learning for predicting urban pluvial flood water depth. *J. Hydrol.* **2021**, *603*, 126898. [CrossRef]

36. Seleem, O.; Ayzel, G.; Bronstert, A.; Heistermann, M. Transferability of data-driven models to predict urban pluvial flood water depth in Berlin, Germany. *Nat. Hazards Earth Syst. Sci.* **2023**, *23*, 809–822. [CrossRef]
37. do Lago, C.A.; Giacomoni, M.H.; Bentivoglio, R.; Taormina, R.; Gomes, M.N.; Mendiondo, E.M. Generalizing rapid flood predictions to unseen urban catchments with conditional generative adversarial networks. *J. Hydrol.* **2023**, *618*, 129276. [CrossRef]
38. Zhao, G.; Pang, B.; Xu, Z.; Peng, D.; Zuo, D. Urban flood susceptibility assessment based on convolutional neural networks. *J. Hydrol.* **2020**, *590*, 125235. [CrossRef]
39. Tien Bui, D.; Hoang, N.-D.; Martínez-Álvarez, F.; Ngo, P.-T.T.; Hoa, P.V.; Pham, T.D.; Samui, P.; Costache, R. A novel deep learning neural network approach for predicting flash flood susceptibility: A case study at a high frequency tropical storm area. *Sci. Total Environ.* **2020**, *701*, 134413. [CrossRef]
40. Seleem, O.; Ayzel, G.; de Souza, A.C.T.; Bronstert, A.; Heistermann, M. Towards urban flood susceptibility mapping using data-driven models in Berlin, Germany. *Geomat. Nat. Hazards Risk* **2022**, *13*, 1640–1662. [CrossRef]
41. Kabir, S.; Patidar, S.; Xia, X.; Liang, Q.; Neal, J.; Pender, G. A deep convolutional neural network model for rapid prediction of fluvial flood inundation. *J. Hydrol.* **2020**, *590*, 125481. [CrossRef]
42. Lin, Q.; Leandro, J.; Wu, W.; Bhola, P.; Disse, M. Prediction of Maximum Flood Inundation Extents with Resilient Backpropagation Neural Network: Case Study of Kulmbach. *Front. Earth Sci.* **2020**, *8*, 332. [CrossRef]
43. Shi, X.; Chen, Z.; Wang, H.; Yeung, D.-Y.; Wong, W.; Woo, W. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. *arXiv* **2015**, arXiv:1506.04214.
44. Ayzel, G.; Heistermann, M.; Sorokin, A.; Nikitin, O.; Lukyanova, O. All convolutional neural networks for radar-based precipitation nowcasting. *Procedia Comput. Sci.* **2019**, *150*, 186–192. [CrossRef]
45. Ravuri, S.; Lenc, K.; Willson, M.; Kangin, D.; Lam, R.; Mirowski, P.; Fitzsimons, M.; Athanassiadou, M.; Kashem, S.; Madge, S.; et al. Skilful precipitation nowcasting using deep generative models of radar. *Nature* **2021**, *597*, 672–677. [CrossRef]
46. Ma, X.; Dai, Z.; He, Z.; Ma, J.; Wang, Y.; Wang, Y. Learning Traffic as Images: A Deep Convolutional Neural Network for Large-Scale Transportation Network Speed Prediction. *Sensors* **2017**, *17*, 818. [CrossRef] [PubMed]
47. Bao, J.; Liu, P.; Ukkusuri, S.V. A spatiotemporal deep learning approach for citywide short-term crash risk prediction with multi-source data. *Accid. Anal. Prev.* **2019**, *122*, 239–254. [CrossRef] [PubMed]
48. Zhang, J.; Zheng, Y.; Sun, J.; Qi, D. Flow Prediction in Spatio-Temporal Networks Based on Multitask Deep Learning. *IEEE Trans. Knowl. Data Eng.* **2020**, *32*, 468–478. [CrossRef]
49. Rosenblatt, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychol. Rev.* **1958**, *65*, 386–408. [CrossRef]
50. LeCun, Y.; Boser, B.; Denker, J.S.; Henderson, D.; Howard, R.E.; Hubbard, W.; Jackel, L.D. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Comput.* **1989**, *1*, 541–551. [CrossRef]
51. LeCun, Y.; Bengio, Y. Convolutional networks for images, speech, and time-series. In *The Handbook of Brain Theory and Neural Networks*; Arbib, M.A., Ed.; MIT Press: Cambridge, MA, USA, 1995.
52. Bengio, Y.; Simard, P.; Frasconi, P. Learning long-term dependencies with gradient descent is difficult. *IEEE Trans. Neural Netw.* **1994**, *5*, 157–166. [CrossRef]
53. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
54. Scarselli, F.; Gori, M.; Tsoi, A.C.; Hagenbuchner, M.; Monfardini, G. The graph neural network model. *IEEE Trans. Neural Netw.* **2009**, *20*, 61–80. [CrossRef]
55. Defferrard, M.; Bresson, X.; Vandergheynst, P. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. *Adv. Neural Inf. Process. Syst.* **2016**, arXiv:1606.09375.
56. Kipf, T.N.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv* **2016**, arXiv:1609.02907.
57. Yu, B.; Yin, H.; Zhu, Z. Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting. *arXiv* **2017**, arXiv:1709.04875.
58. Hajgató, G.; Gyires-Tóth, B.; Paál, G. Reconstructing nodal pressures in water distribution systems with graph neural networks. *arXiv* **2021**, arXiv:2104.13619.
59. Fritz, C.; Dorigatti, E.; Rügamer, D. Combining graph neural networks and spatio-temporal disease models to improve the prediction of weekly COVID-19 cases in Germany. *Sci. Rep.* **2022**, *12*, 3930. [CrossRef]
60. Goodfellow, I.J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative Adversarial Networks. *arXiv* **2014**, arXiv:1406.2661. [CrossRef]
61. Mirza, M.; Osindero, S. Conditional Generative Adversarial Nets. *arXiv* **2014**, arXiv:1411.1784.
62. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M.; et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Available online: https://www.tensorflow.org/ (accessed on 25 January 2022).
63. Andersson, H.; Mariegaard, J.; Ridler, M. MIKE IO: Input/Output of MIKE Files in Python. 2022. Available online: https://dhi.github.io/mikeio/ (accessed on 20 January 2023).
64. DHI. *MIKE+: Release 2021 Update 1*; DHI: Hørsholm, Denmark, 2021. Available online: www.mikepoweredbydhi.com (accessed on 18 November 2022).
65. Bezirksregierung Köln. 3D-Messdaten. Available online: https://www.bezreg-koeln.nrw.de/brk_internet/geobasis/hoehenmodelle/3d-messdaten/index.html (accessed on 20 July 2021).

66. Deutsches Institut für Normung e. V. *Entwässerungssysteme außerhalb von Gebäuden-Kanalmanagement*; Beuth Verlag GmbH: Berlin, Germany, 2017.
67. DWA. *Hydraulische Bemessung und Nachweis von Entwässerungssystemen: Arbeitsblatt DWA-A 118*; Deutsche Vereinigung für Wasserwirtschaft, Abwasser und Abfall (DWA): Hennef, Germany, 2006; ISBN 9783939057154.
68. Schmitt, T.G. Ortsbezogene Regenhöhen im Starkregenindexkonzept SRI12 zur Risikokommunikation in der kommunalen Überflutungsvorsorge. *KA Korresp. Abwasser Abfall* **2017**, *64*, 294–300.
69. Meon, G.; Stein, K.; Förster, K.; Riedel, G. *Abschlussbericht zum Forschungsprojekt "Untersuchung starkregengefährdeter Gebiete*; TU Braunschweig: Braunschweig, Germany, 2009.
70. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv* **2015**, arXiv:1502.03167.
71. Nair, V.; Hinton, G.E. Rectified Linear Units Improve Restricted Boltzmann Machines. In Proceedings of the 27th International Conference on International Conference on Machine Learning, Haifa, Israel, 21–24 June 2010; Omnipress: Madison, WI, USA, 2010; pp. 807–814, ISBN 9781605589077.
72. Han, J.; Moraga, C. The influence of the sigmoid function parameters on the speed of backpropagation learning. In *From Natural to Artificial Neural Computation*; Goos, G., Hartmanis, J., Leeuwen, J., Mira, J., Sandoval, F., Eds.; Springer: Berlin/Heidelberg, Germany, 1995; pp. 195–201, ISBN 9783540594970.
73. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2014**, arXiv:1412.6980.
74. Smith, L.N. Cyclical Learning Rates for Training Neural Networks. *arXiv* **2015**, arXiv:1506.01186.
75. Zhao, L.; Song, Y.; Zhang, C.; Liu, Y.; Wang, P.; Lin, T.; Deng, M.; Li, H. T-GCN: A Temporal Graph ConvolutionalNetwork for Traffic Prediction. *IEEE Trans. Intell. Transport. Syst.* **2018**, *21*, 3848–3858. [CrossRef]
76. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
77. Jamali, B.; Bach, P.M.; Cunningham, L.; Deletic, A. A Cellular Automata Fast Flood Evaluation (CA-ffé) Model. *Water Resour. Res.* **2019**, *55*, 4936–4953. [CrossRef]
78. Guo, Z.; Moosavi, V.; Leitão, J.P. Data-driven rapid flood prediction mapping with catchment generalizability. *J. Hydrol.* **2022**, *609*, 127726. [CrossRef]
79. Sun, J.; Xue, M.; Wilson, J.W.; Zawadzki, I.; Ballard, S.P.; Onvlee-Hooimeyer, J.; Joe, P.; Barker, D.M.; Li, P.-W.; Golding, B.; et al. Use of NWP for Nowcasting Convective Precipitation: Recent Progress and Challenges. *Bull. Am. Meteorol. Soc.* **2014**, *95*, 409–426. [CrossRef]
80. Shazeer, N.; Cheng, Y.; Parmar, N.; Tran, D.; Vaswani, A.; Koanantakool, P.; Hawkins, P.; Lee, H.; Hong, M.; Young, C.; et al. Mesh-TensorFlow: Deep Learning for Supercomputers. *arXiv* **2018**, arXiv:1811.02084.
81. Serpico, S.B.; Dellepiane, S.; Boni, G.; Moser, G.; Angiati, E.; Rudari, R. Information Extraction from Remote Sensing Images for Flood Monitoring and Damage Evaluation. *Proc. IEEE* **2012**, *100*, 2946–2970. [CrossRef]
82. Mateo-Garcia, G.; Veitch-Michaelis, J.; Smith, L.; Oprea, S.V.; Schumann, G.; Gal, Y.; Baydin, A.G.; Backes, D. Towards global flood mapping onboard low cost satellites with machine learning. *Sci. Rep.* **2021**, *11*, 7249. [CrossRef]
83. Nedergaard Pedersen, A.; Wied Pedersen, J.; Vigueras-Rodriguez, A.; Brink-Kjær, A.; Borup, M.; Steen Mikkelsen, P. The Bellinge data set: Open data and models for community-wide urban drainage systems research. *Earth Syst. Sci. Data* **2021**, *13*, 4779–4798. [CrossRef]