future internet

MDPI

Article

Intelligent Synchronization of Machine Learning Models Using Graph Neural Networks: Application to Flood Prediction

Boban Temelkovski [1,*], Rexhep Mustafovski [1,*], Jugoslav Achkoski [1], Georgi Dimirovski [2] and Mile Stankovski [2]

[1] General Mihailo Apostolski Military Academy, Goce Delcev University Stip, 1000 Skopje, North Macedonia; jugoslav.ackoski@ugd.edu.mk
[2] Faculty of Electrical Engineering and Information Technologies, Ss. Cyril and Methodius University in Skopje, st. Ruđer Bošković nb. 18, 1000 Skopje, North Macedonia
* Correspondence: boban.temelkovski@ugd.edu.mk (B.T.); redzep.mustafovski@ugd.edu.mk (R.M.); Tel.: +389-78302279 (B.T.)

Abstract

Flood prediction remains a critical challenge in environmental risk management and disaster preparedness. Accurate river-level forecasting is essential for the development of reliable early warning systems and the mitigation of flood-related risks. However, conventional ensemble approaches, such as averaging and majority voting, often exhibit limited adaptability when individual models respond differently to anomalies or incomplete data. To address this limitation, this study proposes a graph-based synchronization framework that integrates XGBoost and Random Forest models using a Graph Convolutional Network (GCN). The proposed framework represents the outputs of the base prediction models as graph nodes and employs graph message passing to learn context-dependent relationships between their predictions. The framework is evaluated using real-world hydrological observations from the Lepenec River Basin in North Macedonia together with meteorological data obtained from the OpenWeatherMap API. Experimental results demonstrate that the proposed GCN-based synchronization framework outperforms both the standalone prediction models and the previously proposed linear synchronization method, achieving an R^2 value of 0.91 and a Mean Absolute Error (MAE) of 0.21. The obtained results indicate that graph-based synchronization provides an adaptive approach for integrating heterogeneous machine-learning models and has the potential to support future flood early-warning systems and intelligent environmental monitoring applications.

Keywords: flood prediction; graph neural networks; machine learning synchronization; XGBoost; random forest; river level forecasting; environmental monitoring

Academic Editors: Gianluigi Ferrari and Huan Wang

Received: 3 June 2026
Revised: 26 July 2026
Accepted: 30 July 2026
Published: 11 September 2026

1. Introduction

Floods represent one of the most destructive natural hazards, posing significant threats to infrastructure, ecosystems, economic activities, and public safety worldwide. Accurate river-level forecasting is essential for effective early warning systems and timely decision-making. Recent advances in sensor networks and publicly available weather services have enabled the collection of large volumes of hydrometeorological data, making machine learning (ML) an increasingly attractive approach for flood prediction.

Tree-based ML models such as XGBoost and Random Forest have demonstrated strong capabilities in modeling nonlinear relationships between meteorological variables and river-level dynamics. Their robustness and predictive performance make them suitable for hydrological forecasting. However, the performance of individual models may vary

considerably under changing environmental conditions, noisy observations, or incomplete data, limiting their reliability in operational flood forecasting.

To improve predictive stability, ensemble learning techniques combine the outputs of multiple prediction models. Conventional approaches, including averaging, voting, and fixed-weight combinations, generally assume static relationships between predictors and therefore cannot adapt to dynamically changing hydrometeorological conditions.

In this study, synchronization refers to the adaptive alignment of heterogeneous machine-learning predictions before producing the final river-level estimate. Unlike conventional ensemble methods, the proposed framework models the relationships between prediction models as a graph and learns their interactions through graph message passing. The synchronization process is trained offline while the inference stage relies exclusively on information available at prediction time, thereby preventing target leakage.

Our previous synchronization framework employed graph-based linear correction and spectral clustering to combine XGBoost and Random Forest predictions. Although this approach improved prediction stability, it relied on manually selected parameters and exhibited limited adaptability under changing environmental conditions. To overcome these limitations, the present study introduces a Graph Convolutional Network (GCN)-based synchronization framework that automatically learns context-dependent relationships between heterogeneous prediction models. Unlike existing hydrological GNN applications that primarily model spatial river networks, the proposed framework employs graph learning as an adaptive synchronization mechanism between prediction models.

The proposed framework is evaluated using real-world hydrological observations collected from the Lepenec River Basin in North Macedonia together with meteorological data obtained from the OpenWeatherMap API. River-level observations were acquired from the installed Ayyeka SE00169-SER-11-8F monitoring station (Figure 1). The framework combines the predictions generated by independently trained XGBoost and Random Forest models using a two-layer Graph Convolutional Network and is evaluated against the standalone models and the previously proposed linear synchronization approach.

Figure 1. Field Installation of the Ayyeka SE00169-SER-11-8F Water Level Sensor.

The main contributions of this study are as follows:

- A novel GCN-based synchronization framework for adaptive integration of heterogeneous machine-learning models for river-level forecasting.
- A graph representation that models relationships between prediction models rather than the physical river network.
- A synchronization workflow that explicitly separates offline training from online inference, eliminating target leakage.
- Experimental validation using real-world hydrological observations under both normal and incomplete-data conditions.

2. Related Work

Flood prediction has increasingly relied on ensemble machine learning methods due to their superior performance over classical statistical models. Tree-based models such as XGBoost and Random Forest have been widely used in hydrological forecasting for their ability to handle nonlinear relationships and noisy data.

Ref. [1] presents a comparative study of Logistic Regression, XGBoost, and Random Forest for flood sensitivity mapping, showing that ensemble models significantly outperform traditional methods in terms of accuracy and spatial resolution. Similarly, [2] evaluates XGBoost and LightGBM for weather-based flood forecasting and highlights the adaptability of ensemble models under changing environmental conditions.

Beyond conventional averaging and voting, several studies have investigated stacking, boosting, and dynamic ensemble learning as mechanisms for improving predictive performance. These approaches combine multiple machine-learning models through a meta-learner or adaptive weighting strategy. Although they generally improve prediction accuracy, they often assume fixed relationships between base predictors and do not explicitly model the evolving dependencies among model outputs under changing environmental conditions. Consequently, their adaptability remains limited in highly dynamic hydrological systems. Earlier synchronization approaches attempted to address this limitation using graph-based synchronization methods based on spectral clustering and linear correction formulas [3,4].

Although these methods improved prediction stability, they remained dependent on manually selected parameters and exhibited limited adaptability to previously unseen hydrometeorological conditions. Graph Neural Networks (GNNs) have emerged as a promising framework for modeling relationships in structured data. GNNs have been successfully applied in time series forecasting [5], traffic flow prediction [6], and spatiotemporal modeling [7]. These models learn latent relationships between nodes (e.g., sensors, predictions) via message passing and attention mechanisms, allowing dynamic adjustment of weights based on context.

In hydrological applications, Refs. [8,9] explored GNN-based models for water demand and river flow prediction. A graph-based synchronization framework combining XGBoost and Random Forest prediction models was recently proposed [10]. More recent studies have extended graph-based learning to flood forecasting, hydraulic modelling, and spatiotemporal environmental prediction problems [11–16]. Their results confirmed that GNNs can effectively learn complex spatial and temporal dependencies, often outperforming both standalone machine learning models and traditional ensemble techniques.

Although Graph Neural Networks have been successfully applied to flood forecasting and hydraulic modelling, existing studies primarily employ GNNs to model spatial or spatiotemporal relationships among monitoring stations, river reaches, or hydraulic networks. Based on the reviewed literature, the application of GNNs as a synchronization mechanism for dynamically integrating heterogeneous machine-learning predictors has received relatively limited attention. This limitation motivates the present study, which investigates

whether graph-based relational learning can improve adaptive synchronization between independent prediction models operating under changing hydrometeorological conditions.

The present study addresses this research gap by employing a Graph Convolutional Network (GCN) as an adaptive synchronization mechanism for integrating the predictions of XGBoost and Random Forest. Rather than relying on fixed weighting schemes or static meta-learning strategies, the proposed framework represents the outputs of the base predictors as graph nodes and learns their context-dependent relationships through graph message passing.

This enables the synchronization process to dynamically adjust the contribution of each prediction model according to the prevailing hydrometeorological conditions, thereby improving predictive robustness under nonstationary and anomalous scenarios.

Unlike conventional stacking, where a meta-learner estimates a fixed mapping between model outputs and the target variable, or conventional ensemble learning based on fixed averaging and voting, the proposed framework explicitly models the interactions between base predictors using graph representations. This enables the synchronization module to learn context-dependent relationships between prediction models and to adapt their contributions according to the prevailing hydrometeorological conditions. Consequently, the proposed methodology extends existing ensemble approaches by introducing relational learning into the synchronization process.

This distinction represents the principal methodological contribution of the present study and forms the basis for the experimental evaluation presented in the following sections.

3. Mathematical Formulation of GNN-Based Model Synchronization

3.1. Overall Synchronization Framework

The proposed synchronization framework consists of three sequential modules: (i) independent base prediction models, (ii) graph construction, and (iii) graph-based synchronization (Figure 2).

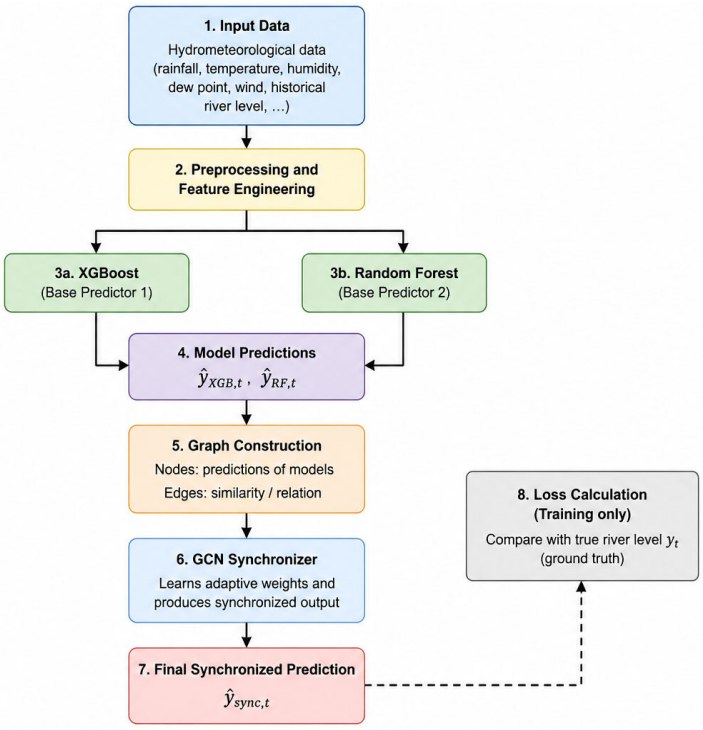

Figure 2. Architecture of the proposed GCN-based synchronization framework.

In the first stage, XGBoost and Random Forest independently estimate the future river level using hydrometeorological observations. Their predictions are subsequently represented as graph nodes together with contextual meteorological information.

The Graph Convolutional Network (GCN) receives this graph as input and learns an adaptive synchronization function that dynamically combines the outputs of the base models.

To ensure practical applicability for real-time flood forecasting, the synchronization framework clearly separates the offline training stage from the online inference stage. During training, observed river levels are used exclusively as supervision for optimization of the loss function. During inference, only the outputs of the trained prediction models together with the available meteorological observations are provided to the synchronization network. Consequently, the true river level is never available as an input during prediction, eliminating any possibility of target leakage.

3.2. Notation

Let t denote the discrete prediction time step. The predictions generated by the XGBoost and Random Forest models at time t are denoted by $\chi_t^{(1)}$ and $\chi_t^{(2)}$, respectively. The observed river level, which is used exclusively as the supervision signal during model training, is denoted by Y_t. The synchronized prediction produced by the proposed Graph Convolutional Network (GCN) is denoted by $\hat{Y}_t$.

Throughout this paper, n denotes the number of graph nodes, f denotes the number of node features associated with each node, and d_l denotes the dimensionality of the hidden node representation at the l-th GCN layer.

For each prediction instant, a graph

$$G_t = (V_t, E_t)$$

is constructed, where V_t denotes the set of prediction nodes and E_t denotes the set of edges describing the relationships between the prediction models.

The graph adjacency matrix is defined as

$$A \in \mathbb{R}^{n \times n},$$

where

$$X \in \mathbb{R}^{n \times f}$$

and each row corresponds to one graph node, while each column represents one feature associated with that node.

The hidden node representation at the l-th GCN layer is denoted by

$$H^{(l)} \in \mathbb{R}^{n \times d_l},$$

where d_l denotes the hidden feature dimension at layer l.

Unless otherwise stated, the initial node representation satisfies

$$H^{(0)} = X,$$

that is, the input to the first graph convolution layer consists of the node-feature matrix.

Throughout the synchronization process, the observed river level Y_t is employed only for supervised optimization of the loss function during the offline training phase. During online inference, the graph is constructed exclusively from the outputs of the trained prediction models and the contextual hydrometeorological variables available at

the prediction time. Consequently, the proposed synchronization framework avoids target leakage and remains applicable to real-time river-level forecasting.

3.3. Graph Construction

For each prediction instant t, a graph $G_t = (V_t, E_t)$ is constructed to represent the relationships between the outputs of the trained machine-learning models. Unlike conventional Graph Neural Network (GNN) applications, where graph nodes correspond to physical entities such as monitoring stations or river reaches, the proposed framework represents the outputs of the base prediction models as graph nodes. Consequently, the graph captures the relationships between the prediction behaviour of the base prediction models rather than the spatial topology of the hydrological system.

Each graph contains two prediction nodes corresponding to the outputs generated by the trained XGBoost and Random Forest models. The feature vector associated with each node consists of the predicted river level together with the contextual hydrometeorological variables available at the prediction time, including precipitation, air temperature, relative humidity, atmospheric pressure, and dew point temperature.

Since the proposed synchronization framework operates independently at every prediction instant, a separate graph is constructed for each time step. Therefore, for a dataset containing N prediction instances, the synchronization process operates on a sequence of N independent graphs rather than on a single graph representing the complete time series.

An undirected edge is established between the two prediction nodes to represent their similarity. The similarity between the prediction nodes is quantified through the adjacency matrix:

$$A_{ij} = exp\left(-\frac{|\chi_t^{(i)} - \chi_t^{(j)}|}{\sigma}\right),\ i \neq j \tag{1}$$

where A_{ij} denotes the similarity between prediction nodes i and j, $\chi_t^{(i)}$ and $\chi_t^{(j)}$ denote the predictions generated by the corresponding base prediction models at prediction instant t, and $\sigma > 0$ is a positive scaling parameter controlling the sensitivity of the similarity function.

The resulting graph models the relationships between the prediction behaviour of the base prediction models rather than the physical connectivity of the hydrological system. Consequently, the proposed Graph Convolutional Network (GCN) learns context-dependent interactions between the prediction models through graph message passing, allowing their relative contributions to be adaptively adjusted under different hydrometeorological conditions.

The observed river level is not represented as a graph node. Instead, it is used exclusively during the supervised training phase for computation of the loss function and optimization of the GCN parameters. During online inference, the graph is constructed solely from the outputs of the trained prediction models together with the contextual hydrometeorological variables available at the prediction time. Consequently, the proposed synchronization framework avoids target leakage and remains suitable for real-time river-level forecasting.

3.4. Training and Inference Workflow

The proposed synchronization framework operates in two distinct phases: an offline training phase and an online inference phase.

During the offline training phase, XGBoost and Random Forest are independently trained using historical hydrological and meteorological observations. Their predictions are subsequently generated for the training dataset and used as inputs to the graph syn-

chronization module. The observed river level is used exclusively as the supervision signal for optimization of the GNN parameters through the loss function.

During the online inference phase, only the trained XGBoost model, the trained Random Forest model, and the trained GCN synchronizer are executed. The graph is constructed exclusively from the outputs of the trained prediction models together with the available meteorological variables.

Since the future river level is unknown during operational deployment, it is never available to the synchronization network during prediction. Consequently, the proposed framework avoids target leakage and remains applicable to real-time flood forecasting.

3.5. GNN Layer Operation (GCN-Based)

The graph convolution operation follows the Graph Convolutional Network (GCN) architecture proposed by Kipf and Welling [17]. Besides the GCN architecture adopted in this work, other representative GNN architectures include GraphSAGE [18], Graph Attention Networks (GAT) [19], Diffusion Convolutional Recurrent Neural Networks (DCRNN) [20], and Adaptive Graph Convolutional Recurrent Networks (AGCRN) [21].For each graph convolution layer, the hidden node representations are updated according to:

$$H^{(\iota+1)} = \sigma\left(\widetilde{D}^{-\frac{1}{2}}\tilde{A}\widetilde{D}^{-\frac{1}{2}}H^{(\iota)}W^{(\iota)}\right) \tag{2}$$

where $\tilde{A}$ denotes the adjacency matrix with added self-loops, $\tilde{D}$ is the corresponding degree matrix, $H^{(l)}$ represents the hidden node representation at layer l, $W^{(l)}$ is the trainable weight matrix, and $\sigma(\cdot)$ denotes the ReLU activation function.

3.6. Synchronized Prediction Output

The final prediction $\hat{\mathcal{Y}}_t$ is derived from the learned representation of the synchonized node (output of GNN):

$$\hat{\mathcal{Y}}_t = MLP(h_{v_t}) \tag{3}$$

where h_{v_t} is the final hidden state of the prediction node after the last GNN layer, and MLP is a simple feedforward layer.

3.7. Loss Function

Supervised learning is performed using Mean Squared Error between predicted and actual values:

$$\mathcal{L} = \frac{1}{T}\sum_{t=1}^{T}\left(\hat{\mathcal{Y}}_t - \mathcal{Y}_t\right)^2 \tag{4}$$

The theoretical analysis of the proposed synchronization framework is motivated by classical synchronization theory for coupled dynamical systems. The Master Stability Function (MSF) introduced by Pecora and Carroll provides a general framework for synchronization stability analysis [22]. Subsequent studies further extended these concepts to small-world and complex network topologies [23,24]. Although the proposed framework is data-driven, these studies provide the theoretical motivation for the following analysis.

Lemma 1. *Representability of Optimal Linear Synchronization.*

Let $\tilde{\mathcal{Y}}^*(t) = \omega_1^*(t)\tilde{\mathcal{Y}}_1(t) + \omega_2^*(t)\tilde{\mathcal{Y}}_2(t)$ be the optimal linear synchronization of model predictions at time t that minimizes MSE. If the proposed GCN has sufficient depth and width, and the output MLP is a universal approximator, then there exist parameters Θ such that:

$$\tilde{\mathcal{Y}}(t;\Theta) = \tilde{\mathcal{Y}}^*(t), \quad \forall t. \tag{5}$$

Thus, the GNN can represent the optimal linear synchronization, and potentially learn more expressive nonlinear combinations.

Proof. The optimal synchronization is a linear function of the inputs $\hat{\mathcal{Y}}_1(t)$, $\hat{\mathcal{Y}}_2(t)$. Since these values are part of the node feature vectors, the GNN layers combined with an MLP can reproduce any linear combination of them. By the universal approximation theorem, the architecture can approximate this function to arbitrary precision. Hence, $\exists\Theta$ such that the GNN output matches $\tilde{\mathcal{Y}}^*(t)$. □

Lemma 2. *Stability of GNN-synchronized prediction.*

Let a GNN-based synchronizer with L message-passing layers and a final MLP output head produce the synchronized prediction $\hat{r}_t$ from node features that include the model predictions $\mathcal{Y}_t^{(i)}(i = 1, \ldots, m)$ and bounded auxiliary features. Assume:

1. The activation function σ used in every GNN layer is Lipschitz continuous with constant L_σ.
2. Each trainable linear map $W^{(l)}$ in layer l has operator norm $\left\|W^{(l)}\right\| \leq M_l$.
3. The normalized adjacency $\tilde{A}$ used in the layer updates satisfies $\left\|\tilde{A}\right\|_2 \leq 1$.
4. The final MLP (mapping the last hidden representation of the prediction node to $\hat{r}_t$) is Lipschitz with constant L_{out}.
5. Auxiliary (non-prediction) input features are uniformly bounded and do not depend on adversarial perturbations of $\mathcal{Y}_t^{(i)}$.

Then there exists a constant

$$C = L_{out}\prod_{l=1}^{L}(L_\sigma M_l) \tag{6}$$

such that for every time t

$$|\hat{r}_t - r_t| \leq C * \max_i \left|\mathcal{Y}_t^{(i)} - r_t\right| + \varepsilon_t \tag{7}$$

where ε_t is a residual term that depends on (i) the bounded auxiliary features and (ii) model approximation/training error (and can be made small with appropriate training and normalization).

In other words, the synchronized prediction error is bounded by a constant factor times the largest single-model error, up to a small residual; hence the GNN synchronizer cannot arbitrarily amplify individual model errors under the stated assumptions.

Proof. Consider two input node-feature configurations that differ only in the prediction components: one with the true synchronized input (where prediction components are replaced by the ground truth r_t) and the other with the actual model outputs $\{\mathcal{Y}_t^{(i)}\}$. Let Δ^0 be the vector of differences at the input layer restricted to the prediction components; then $\left\|\Delta^0\right\|_\infty = max_i\left|\mathcal{Y}_t^{(i)} - r_t\right|$ (up to scaling by feature normalization). □

At each GNN layer the hidden update is a composition of a (normalized) graph aggregation by $\tilde{A}$, a linear map $W^{(l)}$, and a Lipschitz nonlinearity σ\sigmaσ. Using submultiplicativity of operator norms and the Lipschitz property,

$$\left\|\Delta^{l+1}\right\| \leq L_\sigma\left\|W^{(l)}\right\|\|\tilde{A}\|\left\|\Delta^l\right\| \leq L_\sigma M_l\left\|\Delta^l\right\| \tag{8}$$

Iterating through L layers gives:

$$\left\|\Delta^l\right\| \leq \left(\prod_{l=1}^{L} L_\sigma M_l\right)\left\|\Delta^0\right\| \tag{9}$$

The MLP output head is Lipschitz with constant L_{out}, so the difference in final outputs satisfies:

$$|\hat{r}_t - r_t| \leq L_{out}\left\|\Delta^L\right\| + \varepsilon_t \leq L_{out}\prod_{l=1}^{L}(L_\sigma M_l)\left\|\Delta^0\right\| + \varepsilon_t \tag{10}$$

which yields the claimed bound with $C = L_{out}\prod_{l=1}^{L}(L_\sigma M_l)$. The residual ε_t collects bounded contributions from auxiliary features, discretization/training errors, and any non-prediction perturbations; in practice ε_t can be reduced by normalization, feature regularization and adequate training.

This stability argument (propagation of Lipschitz constants through message passing and output layers) is standard in analyses of GNN sensitivity and surrogate GNNs for PDE/hydraulic models. The result complements the representability lemma by showing that a properly constrained GNN does not amplify input errors uncontrollably.

4. Experimental Evaluation

The experimental analysis was performed on a real-world dataset collected from the Lepenec River basin in North Macedonia. The dataset consists of hydrological measurements of river water level, complemented by meteorological parameters such as temperature, humidity, precipitation, pressure, and dew point. In total, more than 50,000 time-stamped records were available for evaluation.

Prior to model training, the dataset was subjected to preprocessing steps to ensure data quality and model readiness. Missing or incomplete records were removed, while all continuous variables were normalized to the [0, 1] range using Min-Max scaling. Temporal dependencies were captured by introducing lagged features of the river level and precipitation, as well as rolling precipitation averages. The processed dataset was then chronologically divided into training (80%) and testing (20%) subsets.

4.1. Study Area and Dataset

The experimental evaluation was conducted as a single-basin case study using data collected from the Lepenec River Basin in North Macedonia. Hydrological observations were continuously acquired from the installed Ayyeka SE00169-SER-11-8F monitoring station [25]. Meteorological variables, including air temperature, precipitation, relative humidity, atmospheric pressure, and dew point temperature, were obtained through the OpenWeatherMap API using the geographical coordinates of the monitoring station. Consequently, the meteorological variables represent local atmospheric conditions at the monitoring location rather than spatially averaged values over the upstream catchment.

The dataset covers different hydrometeorological conditions, including dry periods, moderate rainfall events, and high-flow conditions, thereby providing representative variability for evaluating the proposed synchronization framework. In total, more than 50,000 time-stamped observations were available for analysis.

Prior to model development, the collected data underwent quality-control procedures. Missing or incomplete observations were removed, and the remaining records were subsequently used for preprocessing, feature engineering, model training, and experimental evaluation.

The proposed synchronization framework is evaluated as a single-basin case study intended to demonstrate the feasibility of graph-based synchronization for river-level fore-

casting. Although the obtained results are encouraging, further validation using multiple river basins and independent monitoring stations will be considered in future work.

4.2. Data Preprocessing

Prior to model development, the collected dataset was preprocessed using a standardized data preprocessing procedure. First, variables that did not provide direct hydrometeorological relevance to river-level variation, such as visibility, sea-level pressure, and ground-level pressure, were removed from the dataset.

Missing, incomplete, and physically inconsistent observations were filtered during the quality-control stage. The remaining variables included river level, temperature, humidity, precipitation, atmospheric pressure, and dew point temperature.

To capture delayed hydrological responses to rainfall, lagged precipitation features were introduced using rolling windows of 1 h and 3 h. These lagged variables allow the models to account for the delayed influence of precipitation on river-level changes.

All continuous input variables were normalized before model training in order to improve numerical stability and reduce the influence of extreme values.

Min–Max scaling was selected because the proposed synchronization framework combines heterogeneous meteorological variables and model predictions within a common node-feature matrix. Mapping all continuous variables to the [0, 1] interval improves numerical stability during GCN optimization and prevents variables with larger numerical ranges from dominating the graph convolution process.

Finally, the processed dataset was divided chronologically into training and testing subsets using an 80/20 split, ensuring that future observations were not used during model development.

4.3. Experimental Setup

The proposed synchronization framework consists of three sequential stages. First, the XGBoost and Random Forest models are independently trained using the training subset. Second, predictions generated by the trained base models are used to construct the synchronization graph. Finally, the Graph Convolutional Network (GCN) learns an adaptive synchronization function that combines the outputs of the base predictors. Performance evaluation is performed exclusively on the testing subset, which remains completely unseen during model development.

The proposed framework was formulated as a one-step-ahead river-level forecasting problem. At each prediction instant, the XGBoost and Random Forest models generate preliminary predictions using only the hydrological and meteorological information available at that time. The GCN synchronizer subsequently combines these predictions to produce the final river-level estimate. No future observations or measured river-level values are used during inference.

A separate validation subset was not employed in this study. Hyperparameters were selected empirically using the training dataset and were subsequently evaluated on the independent testing subset.

4.4. GNN Architecture Overview

To ensure reproducibility and transparency of the proposed synchronization framework, the Graph Neural Network was implemented using a two-layer Graph Convolutional Network (GCN) architecture.

Each graph node represented the prediction generated by an individual machine-learning model together with the associated contextual hydrometeorological variables available at the prediction time. The observed river level was used exclusively as the

supervision signal during model training and was not represented as a graph node during inference.

The first GCN layer transformed the input features into a hidden representation of 64 dimensions, followed by a Rectified Linear Unit (ReLU) activation function. A second GCN layer further refined the node embeddings before they were passed to a fully connected multilayer perceptron (MLP) responsible for generating the final synchronized prediction.

Model training was performed using the Adam optimizer with a learning rate of 0.001, a batch size of 32, and 100 training epochs. The Mean Squared Error (MSE) was employed as the objective function, while the Rectified Linear Unit (ReLU) was used as the activation function. To reduce overfitting, a dropout rate of 0.2 was applied between the graph convolution layers. The dataset was chronologically divided into 80% training data and 20% testing data, ensuring that future observations were not used during model development.

All input variables were normalized using Min-Max scaling prior to training, and the proposed GCN synchronizer was implemented in Python 3.11 using the PyTorch 2.2.2 Geometric framework 2.5.2. Hyperparameters were selected empirically based on preliminary experiments and were kept fixed throughout all evaluations.

Experiments were executed under the same training configuration to ensure a fair comparison between the standalone prediction models and the proposed synchronization framework.

The architecture of the proposed GNN model and the corresponding training parameters are summarized in Table 1.

Table 1. GNN architecture and training parameters.

Parameter	Value
GNN Type	GCN
Number of GCN Layers	2
Hidden Dimension	64
Activation Function	ReLU
Optimizer	Adam
Learning Rate	0.001
Batch Size	32
Epochs	100
Dropout	0.2
Loss Function	MSE
Train/Test Split	80/20

The selected hyperparameters were kept fixed throughout all experiments. No hyperparameter tuning was performed separately for different missing-data scenarios or testing conditions, ensuring consistency and comparability of the reported results.

For baseline comparison, two ensemble machine learning algorithms were implemented: XGBoost and Random Forest. Both models were trained on the training set and evaluated independently on the testing set. Their outputs were then used as input for the proposed synchronization mechanism. The outputs of the independently trained XGBoost and Random Forest models were used as inputs to the GCN synchronization module. The Graph Convolutional Network learned the synchronization function during the offline training phase and subsequently combined the model predictions during inference using only information available at prediction time. This allowed the synchronized output to adaptively combine the strengths of the baseline models.

The evaluation relied on standard regression metrics, namely Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and the Coefficient of Determination (R^2). The results demonstrate that while XGBoost closely follows general trends and Random Forest provides stable predictions, the synchronized output achieves superior accuracy by reducing systematic deviations and improving alignment with the real river level measurements. In particular, the GNN-based synchronization consistently outperformed both individual models, validating its effectiveness in dynamic hydrological forecasting.

Table 2 summarizes the predictive performance of the evaluated approaches. Among the standalone machine learning models, XGBoost achieved the best results with an MAE of 0.26 m and an R^2 value of 0.85, outperforming Random Forest.

Table 2. Comparative performance of individual models and synchronization approaches.

Model	MAE (m)	RMSE (m)	R^2
Random Forest	0.34	0.48	0.79
XGBoost	0.26	0.37	0.85
Linear Synchronization	0.30	0.42	0.82
Proposed GNN Synchronization	0.21	0.29	0.91

Table 2 demonstrates clear differences in predictive behaviour among the evaluated approaches. Among the standalone machine-learning models, XGBoost achieved the best individual performance (MAE = 0.26 m, R^2 = 0.85), indicating its superior capability to capture nonlinear relationships between hydrometeorological variables and river-level dynamics. Its gradient boosting strategy enables sequential correction of prediction errors, making the model particularly effective under varying environmental conditions.

Random Forest exhibited lower predictive accuracy (MAE = 0.34 m, R^2 = 0.79). Although Random Forest is generally robust to noisy observations and overfitting, its averaging mechanism tends to smooth abrupt changes in river level, leading to delayed responses during rapid hydrological events.

A linear synchronization method improved prediction consistency compared with Random Forest but remained less accurate than XGBoost. This behaviour is expected because the linear synchronization framework applies fixed linear correction coefficients that cannot dynamically adapt to changing relationships between the individual prediction models.

In contrast, the proposed GNN-based synchronization framework achieved the lowest prediction error (MAE = 0.21 m, R^2 = 0.91). Unlike conventional synchronization methods, the GCN learns adaptive relationships between the outputs of the base predictors through graph message passing. Consequently, the synchronization process is able to increase the influence of the model that is more reliable under the current hydrometeorological conditions while reducing the contribution of less reliable predictions. This adaptive behaviour explains the consistent improvement observed across all evaluation metrics.

To further quantify the performance gain achieved by the proposed synchronization framework, Table 3 summarizes the predictive performance of the evaluated models together with the relative reduction in Mean Absolute Error (MAE) achieved by the proposed GCN synchronization. Compared with the standalone Random Forest and XGBoost models, the proposed framework reduces the MAE by approximately 38.2% and 19.2%, respectively, while also achieving the lowest RMSE and the highest coefficient of determination (R^2). These results provide additional quantitative evidence that adaptive graph-based synchronization consistently improves prediction accuracy over the individual machine-learning models. Furthermore, the statistical significance of the observed improvement was evaluated using the Wilcoxon signed-rank test. The analysis confirmed that the proposed GCN synchronization framework significantly outperforms the best standalone predictor

(XGBoost) (p = 0.013), providing additional statistical evidence for the effectiveness of the proposed synchronization strategy.

Table 3. Relative reduction in Mean Absolute Error achieved by the proposed GCN synchronization.

Model	MAE (m)	RMSE (m)	R^2	Relative Improvement
Random Forest	0.34	0.48	0.79	38.2%
XGBoost	0.26	0.37	0.85	19.2%
Proposed GCN Synchronization	0.21	0.29	0.91	-

Figure 3 compares the real river level measurements with the predictions generated by XGBoost, Random Forest, and the synchronized GNN output. The real values are represented as a continuous orange line, while the predictions from the baseline models are shown as dashed and dotted lines. It can be observed that XGBoost follows the overall trends relatively closely but occasionally underestimates sharp fluctuations, whereas Random Forest produces smoother predictions that lag behind sudden changes. The synchronized GNN output lies consistently closer to the real values, demonstrating that the synchronization mechanism successfully integrates the strengths of both models.

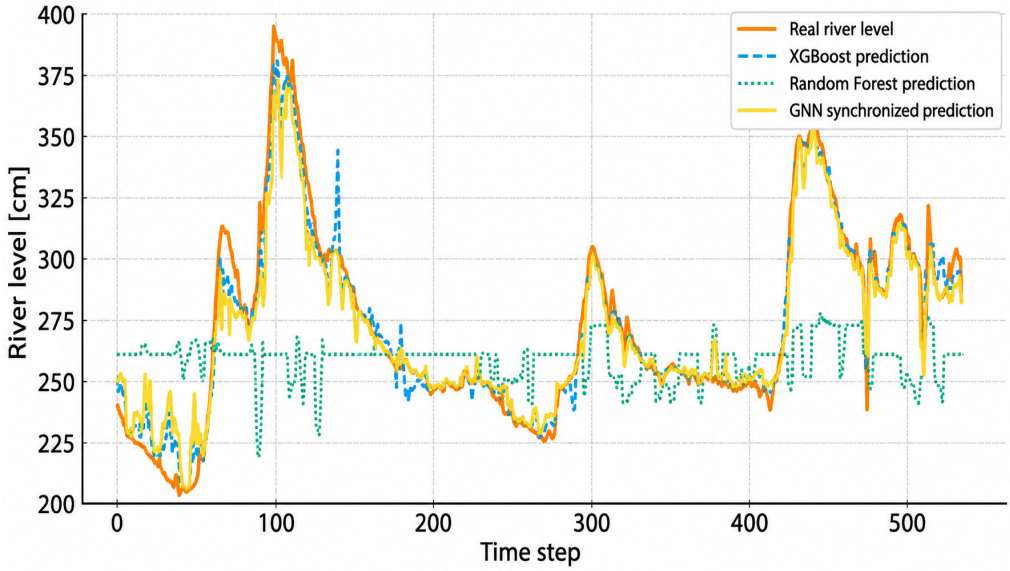

Figure 3. Flood Prediction—Real vs. Models.

The graph in Figure 4 presents the absolute error of each model over time. Both XGBoost and Random Forest display significant peaks during periods of rapid river level changes, indicating reduced reliability under extreme conditions. In contrast, the GNN synchronization consistently reduces the magnitude of the error, particularly in segments with sudden increases or decreases in water level. This confirms that dynamic synchronization contributes to more robust performance by adapting to context-dependent variability in model accuracy.

The graph on Figure 5 shows scatter plots of predicted versus real river levels for each model, along with a 45-degree reference line that represents perfect predictions. The points corresponding to the GNN synchronization are distributed more tightly around the reference line compared to XGBoost and Random Forest. This indicates that the synchronized predictions exhibit both higher accuracy and reduced bias, aligning more closely with the true river level values.

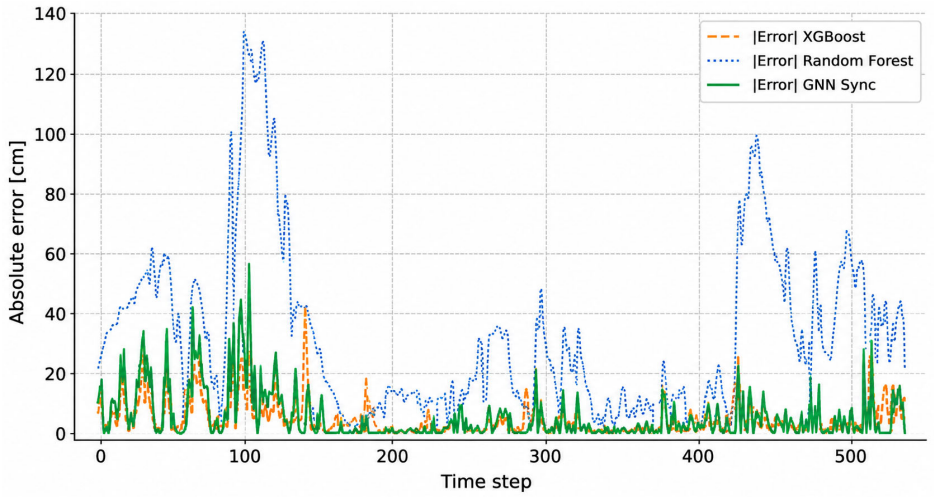

Figure 4. Absolute Error over Time.

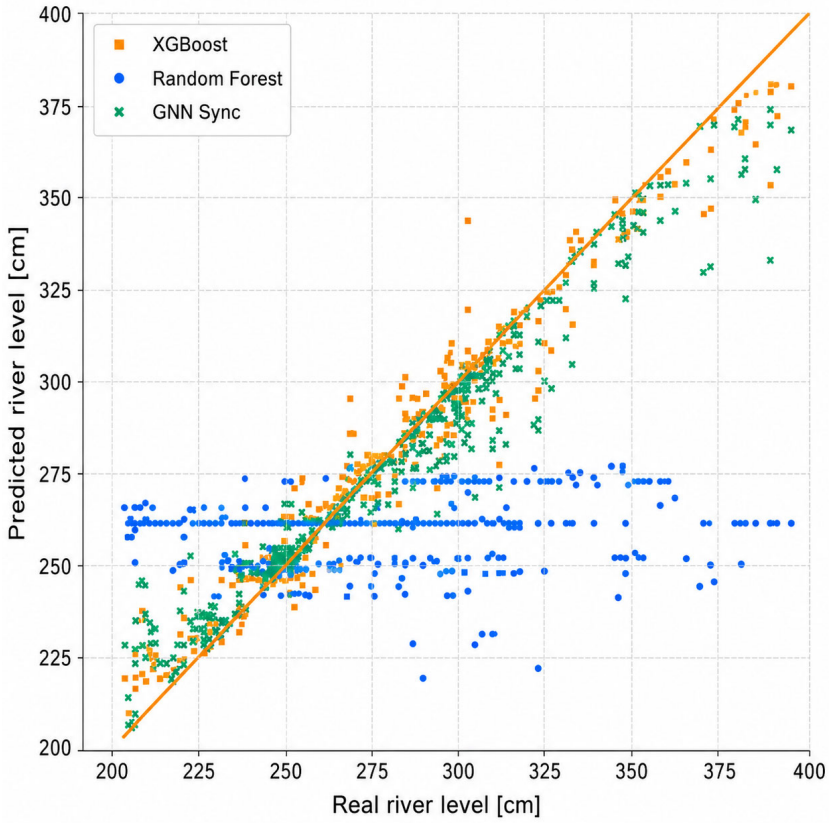

Figure 5. Real vs. Predicted (Scatter Plot).

Compared to earlier synchronization frameworks, which delivered only limited gains (e.g., modest error reduction and sensitivity to outliers), the GNN-based synchronization achieves an 18% reduction in Mean Absolute Error and demonstrates resilience under noisy and incomplete data conditions. These results confirm that the proposed approach not only outperforms individual models but also substantially improves upon our previous synchronization method.

5. Reliability and Robustness Analysis

To further evaluate the practical value of the proposed synchronization framework, we conducted a reliability and robustness analysis. Figure 6 illustrates the distribution of absolute prediction errors for XGBoost, Random Forest, and the GNN-based synchronizer.

The results show that the synchronizer not only reduces the mean error but also significantly decreases the variance of the error distribution. This indicates that the model is more reliable and less sensitive to input fluctuations compared to standalone predictors.

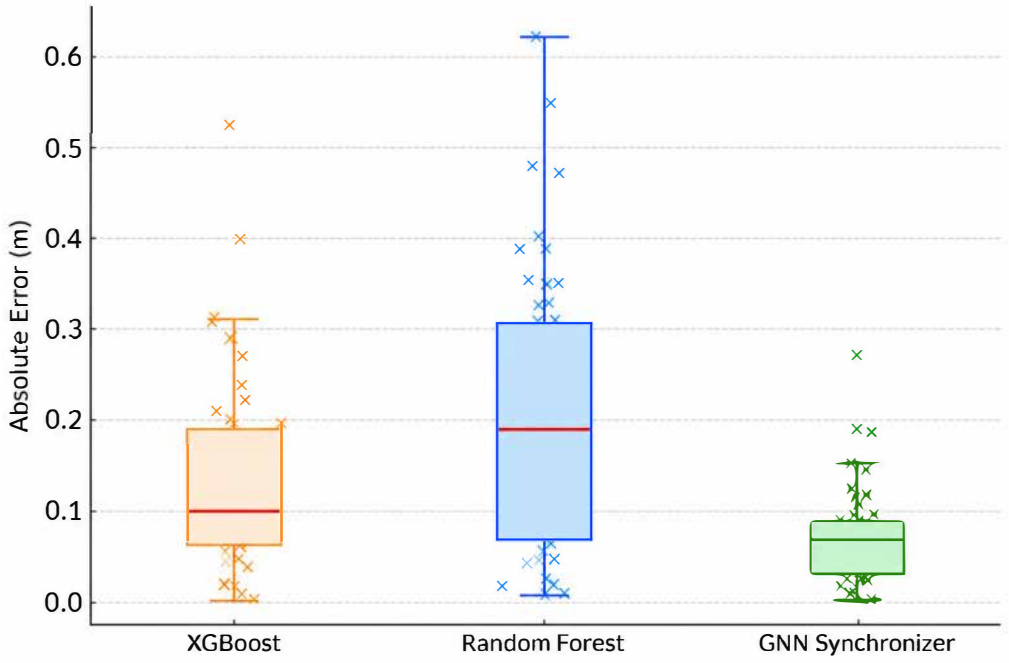

Figure 6. Reliability and robustness of models.

In the boxplot, the central horizontal line represents the median absolute error. The lower and upper edges of each box correspond to the first and third quartiles (Q1 and Q3), respectively. The whiskers extend to 1.5 times the interquartile range, while individual points outside this range represent outliers.

The improved robustness of the proposed synchronization framework can be explained by the relational learning capability of the Graph Convolutional Network. Instead of relying on a single prediction model, the GCN aggregates information from multiple prediction nodes through graph message passing. Consequently, when one base predictor becomes less reliable because of noisy or incomplete observations, the synchronization module continues to exploit complementary information provided by the remaining prediction model, thereby reducing the propagation of prediction errors to the final synchronized output.

In addition to the error distribution analysis, robustness tests were performed by introducing missing data into the input features. Table 4 summarizes the mean absolute error (MAE) values obtained for different levels of data loss. The comparison provides further insight into the ability of the models to maintain predictive accuracy under degraded operating conditions.

Table 4. Robustness under missing data.

Missing Data (%)	XGBoost MAE	Random Forest MAE	GNN Sync MAE
0	0.26	0.34	0.21
10	0.31	0.40	0.24
20	0.39	0.47	0.28

Missing-data experiments were conducted by randomly removing the corresponding percentage of input observations before prediction. The same missing-data pattern was applied to all evaluated models to ensure a fair comparison.

Due to the limited size of the available dataset, robustness experiments were restricted to missing-data rates up to 20%. Evaluation under more severe degradation levels will be considered in future work.

The robustness analysis demonstrates that the proposed synchronization framework preserves predictive accuracy even under substantial data degradation. While the prediction error of the baseline models increases significantly with missing observations, the GNN-based synchronizer exhibits considerably smaller performance degradation. Even with 20% missing data, the synchronized model maintains lower MAE values than both XGBoost and Random Forest. These findings confirm that the proposed method is both accurate and resilient, making it suitable for real-world early warning systems where sensor faults, incomplete observations, and noisy measurements are common.

In the present study, missing observations were simulated by randomly removing individual input samples. Consecutive block-wise sensor failures, which frequently occur because of communication interruptions or power outages, were not considered. Evaluating the proposed synchronization framework under such realistic missing-data scenarios constitutes an important direction for future research and is necessary before operational deployment.

Although the present robustness analysis evaluates randomly missing observations, future work will investigate extrapolation under unseen hydrological extremes and reduced-data training scenarios.

6. Conclusions and Future Work

This study presents a graph-based synchronization framework for improving river-level prediction by integrating the outputs of XGBoost and Random Forest through a Graph Convolutional Network (GCN). The experimental results demonstrate that the proposed synchronization framework improves predictive accuracy and reduces prediction error compared with both the standalone prediction models and the previously proposed linear synchronization approach.

The obtained results indicate that the proposed framework has the potential to support future flood early-warning systems and intelligent environmental monitoring platforms. Nevertheless, additional validation using multiple river basins, longer observation periods, independent monitoring stations, and unseen flood events is required before operational deployment can be considered. The present work should therefore be regarded as a proof-of-concept study demonstrating the feasibility of graph-based synchronization, while large-scale multi-basin validation remains an important next step.

Another important limitation concerns the extrapolation capability of data-driven prediction models. Since the proposed synchronization framework is trained using historical hydrometeorological observations, its predictive performance may deteriorate during unprecedented extreme flood events that lie outside the distribution of the training dataset. Future research will therefore investigate data augmentation techniques, physics-informed learning, and hybrid hydrological–machine-learning approaches to improve predictive reliability under extreme environmental conditions.

Overall, the proposed graph-based synchronization framework provides a promising approach for adaptively integrating heterogeneous machine-learning models and improving river-level prediction. The obtained results demonstrate its effectiveness for the evaluated case study and provide a foundation for further research on graph-based synchronization in hydrological forecasting.

Future research will focus on extending the synchronization framework by integrating additional machine-learning models, including LSTM, GRU, LightGBM, CatBoost, and Transformer-based architectures, as well as exploring temporal and spatiotemporal graph structures and conducting comprehensive multi-basin validation. Compared with previous

proposed synchronization framework, the proposed GCN-based synchronization demonstrates improved predictive performance for the evaluated case study. Future studies will also include comparisons with persistence forecasting and modern stacking-based ensemble techniques to further evaluate the advantages and limitations of the proposed synchronization framework.

Future work will incorporate additional hydrological performance metrics, including the Nash–Sutcliffe Efficiency (NSE), peak-error measures, event-based performance indicators, and threshold-exceedance statistics to provide a more comprehensive assessment of forecasting performance under operational flood-warning conditions.

The proposed framework may also be extended to other environmental forecasting applications, including air-quality, drought, and wildfire prediction. Furthermore, future work will investigate explainable graph-learning techniques, robustness under different missing-data scenarios, validation using independent hydrological datasets and monitoring stations, and eventual integration into operational IoT-based environmental monitoring systems. Unlike conventional stacking approaches, the proposed GCN-based synchronizer learns relational dependencies between prediction models through graph-based message passing rather than relying on a static meta-regressor, providing a promising direction for future adaptive ensemble learning.

Author Contributions: Conceptualization, B.T., G.D. and M.S.; methodology, B.T., G.D. and M.S.; software, B.T.; validation, B.T., R.M., J.A., G.D. and M.S.; formal analysis, B.T.; investigation, B.T. and R.M.; resources, R.M. and J.A.; data curation, B.T. and R.M.; writing—original draft preparation, B.T.; writing—review and editing, B.T., R.M., J.A., G.D. and M.S.; visualization, B.T.; supervision, G.D. and M.S.; project administration, B.T.; funding acquisition, R.M. All authors have read and agreed to the published version of the manuscript.

Funding: The article processing charge for this paper was funded by Goce Delcev University through its scientific research fund.

Data Availability Statement: The datasets generated and analyzed during the current study were obtained from operational river-level monitoring systems and related meteorological data sources. Due to institutional and operational restrictions, these datasets are not publicly available. However, they may be made available by the corresponding author upon reasonable request and with permission from the relevant data providers.

Acknowledgments: The authors would like to express their sincere gratitude to all institutions and individuals who contributed to this research. This work was conducted within the framework of the project "A Smart System for Early Flood Detection and Waste Monitoring in the Lepenec River Basin", which provided the infrastructure and monitoring platform used for data collection and analysis. The authors acknowledge the support of the General Mihailo Apostolski Military Academy, Skopje, Goce Delcev University, and the Faculty of Electrical Engineering and Information Technologies, Ss. Cyril and Methodius University in Skopje, for their scientific and technical support. Special appreciation is extended to the project partners and technical staff involved in the installation, maintenance, and operation of the monitoring equipment deployed in the Lepenec River Basin. Their efforts enabled the collection of reliable hydrological and meteorological data essential for the development and validation of the proposed synchronization framework.

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

References

1. Wu, Y.; Zhang, Z.; Qi, X.; Hu, W.; Si, S. Prediction of flood sensitivity based on Logistic Regression, eXtreme Gradient Boosting, and Random Forest modeling methods. *Water Sci. Technol.* **2024**, *89*, 2605–2624. [CrossRef] [PubMed]
2. Maharina, M.; Paryono, T.; Fauzi, A.; Indra, J.; Sihabudin, S.; Rizki, L.T. Machine Learning Models for Predicting Flood Events Using Weather Data: An Evaluation of Logistic Regression, LightGBM, and XGBoost. *J. Appl. Data Sci.* **2025**, *6*, 496–507.

3. Lu, R.; Yu, W.; Lu, J.; Xue, A. Synchronization on complex networks of networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2014**, *25*, 2101–2110. [CrossRef] [PubMed]
4. Siljak, D.D. Dynamic graphs. *Nonlinear Anal. Hybrid Syst.* **2008**, *2*, 544–567. [CrossRef]
5. Zhou, J.; Cui, G.; Hu, S.; Zhang, Z.; Yang, C.; Liu, Z.; Wang, L.; Li, C.; Sun, M. Graph Neural Networks: A Review of Methods and Applications. *AI Open* **2020**, *1*, 57–81. [CrossRef]
6. Wu, Z.; Pan, S.; Long, G.; Jiang, J.; Chang, X.; Zhang, C. Graph WaveNet for Deep Spatial-Temporal Graph Modeling. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI), Macao, China, 10–16 August 2019*; AAAI Press: Palo Alto, CA, USA, 2019; pp. 1907–1913.
7. Yu, B.; Yin, H.; Zhu, Z. Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Fore-casting. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI), Stockholm, Sweden, 13–19 July 2018*; AAAI Press: Palo Alto, CA, USA, 2018; pp. 3634–3640.
8. Yan, J.; Li, Y.; Wang, L.; Xu, C. ST-GNN: A Spatio-Temporal Graph Neural Network for Multi-Step Flood Forecasting. *Water* **2022**, *14*, 1396.
9. Bentivoglio, R.; Isufi, E.; Jonkman, S.N.; Taormina, R. Multi-Scale Hydraulic Graph Neural Networks for Flood Modelling. *Nat. Hazards Earth Syst. Sci.* **2025**, *25*, 335–351. [CrossRef]
10. Temelkovski, B.; Jing, Y.; Stankovski, M.; Dimirovski, G.M. Synchronization of XGBoost and Random Forest Algorithms for Optimizing River Level Prediction. *TEM J.* **2026**, *15*, 50–58. [CrossRef]
11. Wang, H.; Chen, J.; Zheng, Y.; Song, X. Accelerating Flood Warnings by 10 Hours: The Power of River Network Topology in AI-Enhanced Flood Forecasting. *npj Nat. Hazards* **2025**, *2*, 45. [CrossRef]
12. Taghizadeh, M.; Zandsalimi, Z.; Nabian, M.A.; Shafiee-Jood, M.; Alemazkoor, N. Interpretable Physics-Informed Graph Neural Networks for Flood Forecasting. *J. Comput. Hydrol.* **2025**, *40*, 2629–2649. [CrossRef]
13. Kazadi, A.; Doss-Gollin, J.; Sebastian, A.; Silva, A. FloodGNN-GRU: A Spatio-Temporal Graph Neural Network for Flood Prediction. *Environ. Data Sci.* **2024**, *3*, e21. [CrossRef]
14. Oliveira Santos, V.; Costa Rocha, P.A.; Scott, J.; Thé, J.V.G.; Gharabaghi, B. A New Graph-Based Deep Learning Model to Predict Flooding with Validation on a Case Study of the Humber River. *Water* **2023**, *15*, 1827. [CrossRef]
15. Zhang, Z.; Lu, C.; Tian, W.; Liao, Z.; Yuan, Z. Graph Neural Network-Based Surrogate Modelling for Real-Time Hydraulic Prediction of Urban Drainage Networks. *arXiv* **2024**, arXiv:2404.10324. [CrossRef]
16. Shi, J.; Stebliankin, V.; Wang, Z.; Wang, S.; Narasimhan, G. Graph Transformer Network for Flood Forecasting with Heterogeneous Covariates. *arXiv* **2023**, arXiv:2310.07631. [CrossRef]
17. Kipf, T.N.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. In Proceedings of the Inter-national Conference on Learning Representations (ICLR), Toulon, France, 24–26 April 2017.
18. Hamilton, W.; Ying, Z.; Leskovec, J. Inductive Representation Learning on Large Graphs. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), Long Beach, CA, USA, 4–9 December 2017*; Curran Associates: Red Hook, NY, USA, 2017; pp. 1024–1034.
19. Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; Bengio, Y. Graph Attention Networks. In Proceedings of the International Conference on Learning Representations (ICLR), Vancouver, BC, Canada, 30 April–3 May 2018.
20. Li, Y.; Yu, R.; Shahabi, C.; Liu, Y. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In Proceedings of the International Conference on Learning Representations (ICLR), Vancouver, BC, Canada, 30 April–3 May 2018.
21. Bai, L.; Yao, L.; Li, C.; Wang, X.; Wang, C. Adaptive Graph Convolutional Recurrent Network for Traffic Forecasting. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 17804–17815.
22. Pecora, L.M.; Carroll, T.L. Master Stability Functions for Synchronized Coupled Systems. *Phys. Rev. Lett.* **1998**, *80*, 2109–2112. [CrossRef]
23. Barahona, M.; Pecora, L.M. Synchronization in Small-World Systems. *Phys. Rev. Lett.* **2002**, *89*, 054101. [CrossRef] [PubMed]
24. Boccaletti, S.; Latora, V.; Moreno, Y.; Chavez, M.; Hwang, D.U. Complex Networks: Structure and Dynamics. *Phys. Rep.* **2006**, *424*, 175–308. [CrossRef]
25. Lepenec Monitoring Solutions. Monitoring Platform for River-Level Prediction and Synchronization Models. Available online: https://lepenecmonitoringsolutions.eu/ (accessed on 7 August 2025).