Journal of
Hydroinformatics

Journal of Hydroinformatics Vol 28 No 5, 415 doi: 10.2166/hydro.2026.092

Spatio-temporal graph neural networks for the forecast of natural water systems

Kinga Szatmári [a,b,*], Sándor Németh[a], János Abonyi[a,c] and Alex Kummer[a,c]

[a] Department ofProcess Engineering, University of Pannonia, Egyetem st. 10, Veszprém H-8200, Hungary
[b] National Laboratory for Water Science and Water Security, University of Pannonia, Egyetem st. 10, Veszprém H-8200, Hungary
[c] HUN-REN-PE Complex Systems Monitoring Research Group, University of Pannonia, Egyetem st. 10, Veszprém H-8200, Hungary
*Corresponding author. E-mail: szatmari.kinga@mk.uni-pannon.hu

KS, 0009-0000-1102-8234

ABSTRACT

Accurately forecasting natural water systems is a complex task due to their interconnected structure, where both spatial and temporal dependencies play a critical role. In this work, we applied spatio-temporal graph neural networks of varying complexity to forecast the flow of rivers and the total releases of reservoirs in the Upper Colorado River Basin. Since prolonged droughts driven by climate change can reduce water levels in hydrological systems to critical thresholds, it is essential to forecast to mitigate their negative consequences. The models were trained using five years of historical time series data from directly connected sensor points within a river basin. We evaluated six models and compared their forecasting performance using mean squared error, overall, in boxplots. The graph convolutional recurrent network model performed the best compared to the other five models in the case study, which indicates that the graph convolution with the Chebyshev polynomial has the best forecast accuracy in water system forecasting.

Key words: forecasting, spatio-temporal graph neural network, time series, water system

HIGHLIGHTS

- Spatio-temporal graph neural network forecast for total releases in a water system.
- Forecast accuracy comparison of various spatio-temporal graph neural network models.
- Graph convolutional recurrent network outperforms other graph neural network models.

1. INTRODUCTION

In a connected natural water system, a common time series forecasting problem is water flow forecasting from previously observed data at the measurement sensors to predict the future water flow. Forecasting river flows or reservoir levels is crucial for effective water resource management, ensuring sufficient water supply, power generation, and environmental sustainability. Accurate forecasts help stakeholders plan and implement measures to mitigate possible water shortages and adapt to changing conditions, maintaining the balance of the entire water system (Baker *et al.* 2022). For an efficient forecast, graph neural network-based spatio-temporal methods can be applied, where the temporal sequence data and spatial relationships between the measurement points are integrated (Wang *et al.* 2024).

In hydrological systems, various statistical and deep learning models have been applied for time series forecasting. Water monitoring systems provide valuable historical and real-time data on hydrological and environmental conditions, which can be used to identify trends and detect anomalies (Barker *et al.* 2022). These data sets serve as a foundation for the development of forecast models that forecast future water availability, quality fluctuations, and potential risks, improving proactive management strategies (Sha *et al.* 2021).

Spatio-temporal graph neural networks (STGNNs) (Jin *et al.* 2023) are a class of models that integrate spatial relationships and temporal dynamics within a unified framework. By extending graph neural networks (Zhou *et al.* 2020) with sequence learning components, it can capture the evolution of features over time at each node while also considering interactions between nodes in the graph structure. This dual capacity is particularly advantageous for hydrological systems, where both upstream flow conditions and local temporal patterns jointly influence water availability. Spatio-temporal graph neural networks have shown strong performance in environmental applications due to their ability to learn complex dynamic dependencies over space and time, making them a promising tool for accurate and scalable forecasting of river flow, reservoir levels, and other hydrological variables (Akkala *et al.* 2025).

Recent hydrologic applications demonstrate that different STGNN models can be applied based on the graph construction, the temporal layer, or the forecast horizon. In practice, these choices define how spatial dependence is encoded (topology or distance-based, directed or undirected connectivity), how temporal dynamics are captured (GRU/LSTM, attentional, or convolutional layer), and what could be the time step ahead of the forecast (hours to months and years). The assumed connectivity directly controls information propagation across the monitoring network and can influence both physical consistency and predictive skill in basin-scale forecasting.

In the Upper Delaware River Basin, an LSTM-based spatio-temporal GNN (DSTGNN) is used for multi-station runoff prediction at short horizons (3/6/9 hours) with a distance-weighted adjacency matrix, and is benchmarked against several spatio-temporal baselines (Yang *et al.* 2023). In the Yangtze River Basin, next-day streamflow prediction is formulated on an unweighted station graph using a Bayesian edge-conditioned graph convolution to model inter-station relationships and predictive uncertainty (Liu *et al.* 2023). Beyond streamflow, a GRU-based spatio-temporal network has been applied to drought category forecasting using meteorological and evapotranspiration-related inputs with a distance-based adjacency (Yu *et al.* 2024). A GraphWaveNet-style spatio-temporal graph learning with a self-adaptive adjacency matrix has been used for groundwater-level prediction across monitoring wells over longer time horizons (Bai & Tahmasebi 2023). Closest to our basin-scale setting, Akkala *et al.* (2025) apply an STGNN to the Upper Colorado River Basin for monthly streamflow forecasting over multi-month horizons (up to 36 months), benchmark a single STGNN against classical models (SARIMA, random forest, LSTM, GRU), and evaluate a sequence sensitivity analysis.

In our work, we also make forecasts for the Upper Colorado River Basin, but focus on daily, next-day forecasting. We model the river basin as a directed, unweighted graph that includes both river and reservoir monitoring stations as nodes, with edges following physical flow directions encoded in the adjacency matrix. We jointly forecast the total release of reservoirs together with the flow of rivers. Finally, rather than evaluating a single STGNN, we perform a controlled comparison of multiple STGNN variants to quantify how architectural choices (GCN-based or Chebyshev-based spectral graph convolutions) affect basin-scale forecasting performance, providing design evidence for short-horizon, daily operational forecasting tasks.

The novelty of our work is the following.

- We demonstrated that spatio-temporal graph neural networks are well-suited for forecasting in complex hydrological systems, such as in a river basin with 10 river and 10 reservoir measurement points.
- We compared EvolveGCN-H, EvolveGCN-O, temporal graph convolutional network (T-GCN), graph convolutional recurrent network (GCRN), STGCN, attention-based spatial-temporal graph convolutional network (ASTGCN) and LSTM for the forecast of the flow of the river and total release of the reservoirs.
- The accuracy of the graph neural networks was investigated, and the GCRN model performs the best, but the accuracy of STGCN and ASTGCN are also close to the GCRN model.

2. SPATIO-TEMPORAL GRAPH NEURAL NETWORK FORECASTING

Accurate water flow forecasting models are required because decisions based on these forecasts have significant practical, environmental, and economic consequences. Inaccurate forecasts can lead to severe mismanagement of water resources, either by underestimating or overestimating future water availability. Spatio-temporal graph neural networks are particularly advantageous in this context, as they can incorporate both spatial structure and temporal variation into the learning process.

In this study, the water system is modeled as a directed graph, where the spatial relationships between monitoring locations are represented by the edges of the graph, and the temporal characteristic of the hydrological variables is captured through time series in the nodes of the graph. This approach enables the model to learn spatial correlations across the network, allowing for more accurate forecasting of variables. The conventional time series models often treat locations independently, but the spatio-temporal graph neural network architectures assume that one node feature is influenced by the neighboring nodes. The proposed methodology for the forecast in the water system is shown in Figure 1.

In spatio-temporal graph neural networks, a graph neural network layer is applied as a spatial layer, and as a temporal layer, a recurrent neural network is used. Appendix A presents the description of graph neural networks, where the degree matrix, the adjacency matrix, and the message passing mechanism are described. Appendix A.1 and Appendix A.2

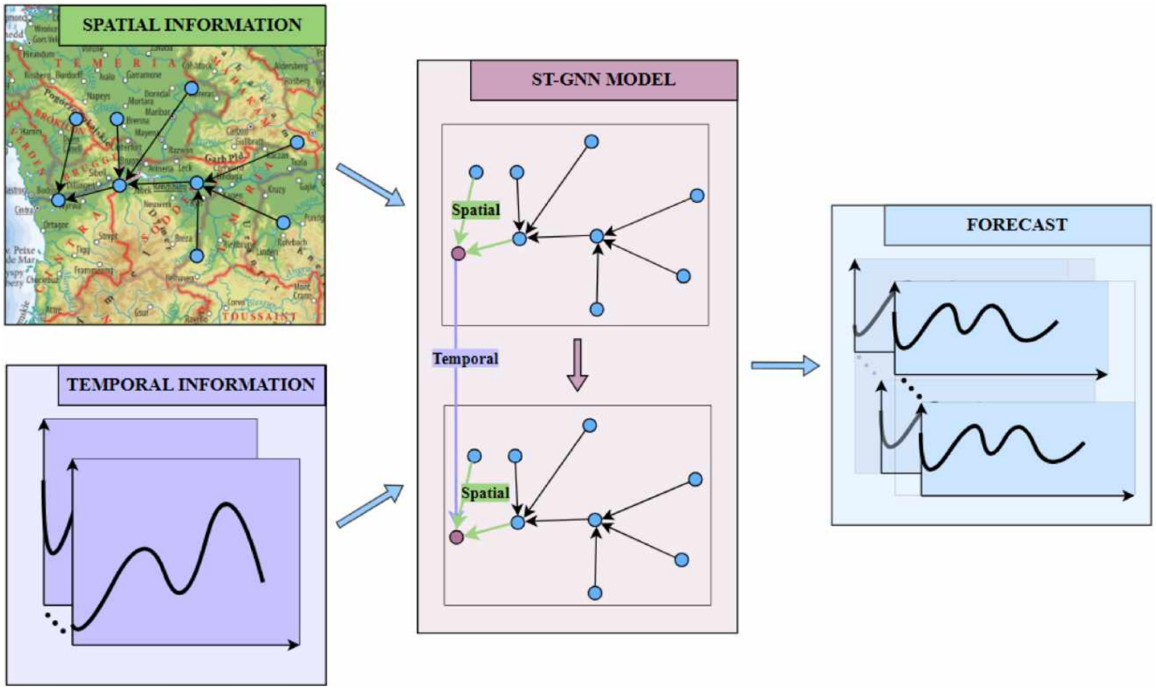

Figure 1 | Spatio-temporal graph neural network model for water system forecasting. In this model, the nodes represent sensor locations, with node features varying over time as temporal information. The spatial information is represented by the edges, which describe the relationships between the sensor locations.

present the two message passing mechanisms, the graph convolution layer and the attentional layer, respectively. The graph convolution layer can be determined by the Chebyshev polynomial and the graph convolutional network (GCN).

In spatio-temporal graph neural networks the spatial and temporal layers have two combinations. One is a modified recurrent neural network, where the input is the node features, and a fully connected layer is replaced by a graph neural network layer in the RNN. The other combination uses a graph neural network to obtain node embedding to learn spatial features, and then a recurrent neural network learns the temporal features of the system (Pareja *et al.* 2020). The six temporal graph neural networks applied in this work are EvolveGCN-H, EvolveGCN-O, T-GCN, GCRN, STGCN, and ASTGCN. In this work, we include these architectures to systematically compare how different spatial-layer designs and spatio-temporal coupling strategies affect basin-scale hydrological forecasting performance. Table 1 summarizes the spatial and temporal blocks of each model, and the structures about the different spatio-temporal graph neural network models are illustrated in Appendix C.

2.1. EvolveGCN

EvolveGCN uses a recurrent model to update the graph convolutional network parameters. The input data consists of the graph adjacency matrix ($\mathbf{A}_t \in \mathbb{R}^{n \times n}$), where n is the number of nodes, t is the times index. The normalization of $\mathbf{A}_t$ is

Table 1 | Comparison of STGNN models based on the temporal and spatial layers

	Spatial layer	Temporal layer
EvolveGCN-H	GCN	GRU
EvolveGCN-O	GCN	LSTM
T-GCN	GCN	GRU
GCRN	Chebyshev	GRU
STGCN	Chebyshev	Convolution
ASTGCN	Attention	Convolution

$\hat{\mathbf{A}}_t \in \mathbb{R}^{n\times n}$, which is defined in Equation (1), where $\widetilde{\mathbf{D}}_t \in \mathbb{R}^{n\times n}$ is the degree matrix as $\widetilde{\mathbf{D}}_t = \mathbf{D}_t + \mathbf{I}_n$, and $\widetilde{\mathbf{A}}_t = \mathbf{A}_t + \mathbf{I}_n$, where $\mathbf{I}_n \in \mathbb{R}^{n\times n}$ is the identity matrix, so self-loops are added to both matrices (Pareja *et al.* 2020).

$$\hat{\mathbf{A}}_t = \widetilde{\mathbf{D}}_t^{-\frac{1}{2}}\widetilde{\mathbf{A}}_t\widetilde{\mathbf{D}}_t^{-\frac{1}{2}} \tag{1}$$

The input also consists of the input node feature matrix ($\mathbf{X}_t \in \mathbb{R}^{n\times d}$), where d is the number of features at each node. The weight matrix $\boldsymbol{\theta}_t \in \mathbb{R}^{d\times d}$ updates the embedding matrix of the node to the output, which is $\mathbf{H}_t \in \mathbb{R}^{n\times d}$. Therefore, the output layer is given by Equation (2), where the sigmoid is the activation function for all but the output layer. For the output layer, it can be considered the identity function or can be the softmax function for node classification (Pareja *et al.* 2020).

$$\mathbf{H}_t = \sigma(\hat{\mathbf{A}}_t\mathbf{X}_t\boldsymbol{\theta}_t) \tag{2}$$

EvolveGCN is schematically visualized in Appendix C in Figure 9, where each time step the model runs a GCN whose weights are supplied by an RNN. The weights are learned by the recurrent neural network, and the M time series is applied from $\mathbf{X}_{t-M}$ to $\mathbf{X}_t$. The GCN weight matrix can be updated in two ways, in the first case $\boldsymbol{\theta}_t$ is treated as the hidden state of the recurrent neural network. The input of the RNN is the embeddings of the nodes, so Equation (3) can be written, and this is called EvolveGCN-H. In the second case, θ_t is the output of the system, as shown in Equation (4), which is called EvolveGCN-O (Pareja *et al.* 2020).

$$\boldsymbol{\theta}_t = \mathrm{RNN}\ (\mathbf{X}_t, \boldsymbol{\theta}_{t-1}) \tag{3}$$

$$\boldsymbol{\theta}_t = \mathrm{RNN}\ (\boldsymbol{\theta}_{t-1}) \tag{4}$$

The two functions with GRU and LSTM are defined in Equations (5) and (6), respectively (Pareja *et al.* 2020).

$$\begin{aligned}[\mathbf{H}_t, \boldsymbol{\theta}_t] &= \mathrm{EvolveGCN\text{-}H}(\mathbf{A}_t, \mathbf{X}_t, \boldsymbol{\theta}_{t-1})\\ \boldsymbol{\theta}_t &= \mathrm{GRU}(\mathbf{X}_t, \boldsymbol{\theta}_{t-1})\\ \mathbf{H}_t &= \mathrm{GCONV}(\mathbf{A}_t, \mathbf{X}_t, \boldsymbol{\theta}_t)\end{aligned} \tag{5}$$

$$\begin{aligned}[\mathbf{H}_t, \boldsymbol{\theta}_t] &= \mathrm{EvolveGCN\text{-}O}(\mathbf{A}_t, \mathbf{X}_t, \boldsymbol{\theta}_{t-1})\\ \boldsymbol{\theta}_t &= \mathrm{LSTM}(\boldsymbol{\theta}_{t-1})\\ \mathbf{H}_t &= \mathrm{GCONV}(\mathbf{A}_t, \mathbf{X}_t, \boldsymbol{\theta}_t)\end{aligned} \tag{6}$$

For EvolveGCN-H, a pooling layer is applied, which is presented in Appendix B, because it can only be used when the column dimension of the input matches the dimension of the hidden state. EvolveGCN-H is usually applied when the node features are informative, since it merges the node embedding to the recurrent network. EvolveGCN-O is applied when the node features are not as informative, but the graph structure plays an important role in prediction (Pareja *et al.* 2020).

2.2. Temporal graph convolutional network

The T-GCN integrates the graph convolutional network and the gated recurrent unit, and $\mathbf{X}_t \in \mathbb{R}^{n\times d}$ is the input node feature matrix at time t. Equation (7) describes a one-layer model, where $\widehat{\mathbf{A}}_t$ is described in Equation (1), $\boldsymbol{\theta}_0 \in \mathbb{R}^{d\times d}$ and is the weight matrix and $ReLU(\cdot)$ is the activation functions (Zhao *et al.* 2019).

$$f(\widehat{\mathbf{A}}_t, \mathbf{X}_t) = Relu(\widehat{\mathbf{A}}_t\mathbf{X}_t\boldsymbol{\theta}_0) \tag{7}$$

T-GCN as described in Equation (8), where $\mathbf{H}_{t-1} \in \mathbb{R}^{n\times d}$ is the output at time $t-1$, $\mathbf{U}_t \in \mathbb{R}^{n\times d}$, $\mathbf{R}_t \in \mathbb{R}^{n\times d}$, and $\mathbf{C}_t \in \mathbb{R}^{n\times d}$ are the update gate, the reset gate and the candidate activation at time t, respectively, $\mathbf{W}_u \in \mathbb{R}^{n\times n\cdot d}$, $\mathbf{W}_r \in \mathbb{R}^{n\times n\cdot d}$, $\mathbf{W}_c \in \mathbb{R}^{n\times n\cdot d}$ are the weight matrices, $\mathbf{b}_u$, $\mathbf{b}_r$, $\mathbf{b}_c$ are the bias vectors, $\mathbf{H}_t$ is the output at time t, and $\odot$ is the Hadamard (element-wise) product, and $\sigma(\cdot)$, $tanh(\cdot)$ are the activation functions (Zhao *et al.* 2019). In Appendix C Figure 10 represents the structure of the

T-GCN, where $\mathbf{X}_{t-M}, \ldots, \mathbf{X}_t$ are the input node feature matrices from time $t-M$ to time t, and $\mathbf{H}_{t-M}, \ldots, \mathbf{H}_t$ are the outputs.

$$\begin{aligned}
\mathbf{U}_t &= \sigma(\mathbf{W}_u[f(\widehat{\mathbf{A}}_t, \mathbf{X}_t), \mathbf{H}_{t-1}] + \mathbf{b}_u) \\
\mathbf{R}_t &= \sigma(\mathbf{W}_r[f(\widehat{\mathbf{A}}_t, \mathbf{X}_t), \mathbf{H}_{t-1}] + \mathbf{b}_r) \\
\mathbf{C}_t &= tanh(\mathbf{W}_c[f(\widehat{\mathbf{A}}_t, \mathbf{X}_t), \mathbf{R}_t \odot \mathbf{H}_{t-1}] + \mathbf{b}_c) \\
\mathbf{H}_t &= \mathbf{U}_t \odot \mathbf{H}_{t-1} + (1 - \mathbf{U}_t) \odot \mathbf{C}_t
\end{aligned} \tag{8}$$

2.3. Graph convolutional recurrent network

The GCRN integrates graph convolution with the Chebyshev polynomial and the GRU (Seo *et al.* 2018). The recurrent neural network is usually GRU, and the model is defined in Equation (9), where the graph convolutional kernel is defined by $\mathbf{W}_{h\cdot} \in \mathbb{R}^{K \times d \times n}$ and $\mathbf{W}_{x\cdot} \in \mathbb{R}^{K \times d \times n}$, which are the Chebyshev coefficients, $\mathbf{H}_{t-1} \in \mathbb{R}^{n \times d}$ is the output at time $t-1$, $\mathbf{U}_t \in \mathbb{R}^{n \times d}$, $\mathbf{R}_t \in \mathbb{R}^{n \times d}$, and $\mathbf{C}_t \in \mathbb{R}^{n \times d}$ are the update gate, the reset gate and the candidate activation at time t, respectively, $\mathbf{H}_t$ is the output at time t, $\odot$ is the Hadamard (element-wise) product, and $\sigma(\cdot)$, $tanh(\cdot)$ are the activation functions (Zhao *et al.* 2019). The structure of GCRN is shown in Appendix C in Figure 11.

$$\begin{aligned}
\mathbf{U}_t &= \sigma(\mathbf{W}_{xu} *_G \mathbf{X}_t + \mathbf{W}_{hu} *_G \mathbf{H}_{t-1}) \\
\mathbf{R}_t &= \sigma(\mathbf{W}_{xr} *_G \mathbf{X}_t + \mathbf{W}_{hr} *_G \mathbf{H}_{t-1}) \\
\mathbf{C}_t &= tanh(\mathbf{W}_{xc} *_G \mathbf{X}_t + \mathbf{W}_{hc} *_G (\mathbf{R}_t \odot \mathbf{H}_{t-1})) \\
\mathbf{H}_t &= \mathbf{U}_t \odot \mathbf{H}_{t-1} + (1 - \mathbf{U}_t) \odot \mathbf{C}_t
\end{aligned} \tag{9}$$

2.4. Spatio-temporal graph convolutional network

The STGCN uses two ST-blocks, where there is a spatial graph convolutional layer between two gated sequential convolution layers. The structure of a block of the spatio-temporal graph (St-block) is described in Appendix C in Figure 12. In this structure, after the two ST-blocks, a temporal convolution layer is applied as the output layer, which maps the output of the second ST-block to a single-step prediction (Yu *et al.* 2017).

The graph convolutional layer is described with the Chebyshev polynomial. The temporal convolutional layer has a 1-D casual convolution with a K_t kernel, and then, for nonlinearity, a gated linear unit (GLU). The neighbors K_t of each node in the input are explored by the temporal convolution, so the length is reduced to $K_t - 1$ each time. The input of the temporal convolution for each node is written as $\mathbf{X}_n \in \mathbb{R}^{n \times d}$. The convolutional kernel $\mathbf{\Gamma} \in \mathbb{R}^{K_t \times d \times 2d}$ maps the input to a single output element, so the temporal gated convolution is described in Equation (10), where $\mathbf{P} \in \mathbb{R}^{n-Kt+1 \times 2d}$ and $\mathbf{Q} \in \mathbb{R}^{n-Kt+1 \times 2d}$ are the inputs of the gates in GLU (Yu *et al.* 2017).

$$\mathbf{\Gamma} *_\tau \mathbf{X}_n = \mathbf{P} \odot \sigma(\mathbf{Q}) \tag{10}$$

In the St-Conv block, the input ($\mathbf{X}^l \in \mathbb{R}^{M \times n \times C^l}$) of block l and the output ($\mathbf{H}^{l+1} \in \mathbb{R}^{(M-2(K_t-1)) \times n \times C^{l+1}}$) are shown in Equation (11), where $\mathbf{\Gamma}_0^l$ and $\mathbf{\Gamma}_1^l$ are the two temporal kernels in the l block, respectively, and W^l is the spectral kernel of graph convolution (Yu *et al.* 2017).

$$\mathbf{H}^{l+1} = \mathbf{\Gamma}_1^l *_\tau ReLU(W^l *_G (\mathbf{\Gamma}_0^l *_\tau \mathbf{X}^l)) \tag{11}$$

2.5. Attention-based spatial-temporal graph convolutional network

An ASTGCN can also be applied to predict flow sequences $\mathbf{H} \in \mathbb{R}^{n \times T_p}$ during the next T_p time slices. The ASTGCN model is shown in Appendix C in Figure 13, where $\mathbf{X}_h \in \mathbb{R}^{n \times d \times T_h}$ represents the present segment, $\mathbf{X}_d \in \mathbb{R}^{n \times d \times T_d}$ represents the daily periodic segment, $\mathbf{X}_w \in \mathbb{R}^{n \times d \times T_w}$ represents the weekly periodic segment, SAtt is spatial attention and TAtt is temporal attention (Guo *et al.* 2019).

Let T_p be the prediction horizon. We divide the historical record into three windows of length T_h, T_d, and T_w (recent, daily-periodic, and weekly-periodic, respectively), each chosen as an integer multiple of T_p. With current time t_0, the recent segment time series is described in Equation (12), which is the just past flow (Guo *et al.* 2019).

$$\mathbf{X}_h = (\mathbf{X}_{t_0-T_h+1}, \mathbf{X}_{t_0-T_h+2}, \ldots, \mathbf{X}_{t_0}) \tag{12}$$

The time series of the daily period ($\mathbf{X}_d \in \mathbf{R}^{n\times d\times T_h}$) is shown in Equation (13), which collects information from the last days, the same amount of period as the prediction period. The current time is denoted as t_0, q is the sampling frequency per day, T_p is the size of the predicting window, and T_d is the size of the daily-periodic segment (Guo *et al.* 2019).

$$\begin{aligned}\mathbf{X}_d = (&\mathbf{X}_{t_0-(T_d/T_p)*q+1}, \ldots, \mathbf{X}_{t_0-(T_d/T_p)*q+T_p},\\ &\mathbf{X}_{t_0-(T_d/T_p-1)*q+1}, \ldots, \mathbf{X}_{t_0-(T_d/T_p-1)*q+T_p},\\ &\mathbf{X}_{t_0-q+1}, \ldots, \mathbf{X}_{t_0-q+T_p})\end{aligned} \tag{13}$$

The weekly period time series ($\mathbf{X}_w \in \mathbf{R}^{n\times d\times T_w}$) is shown in Equation (14), collects information from the last few weeks in the same time period as the prediction period, where T_w is the size of the weekly-periodic segment (Guo *et al.* 2019).

$$\begin{aligned}\mathbf{X}_w = (&\mathbf{X}_{t_0-7*(T_w/T_p)*q+1}, \ldots, \mathbf{X}_{t_0-7*(T_w/T_p)*q+T_p},\\ &\mathbf{X}_{t_0-7*(T_w/T_p-1)*q+1}, \ldots, \mathbf{X}_{t_0-7*(T_w/T_p-1)*q+T_p},\\ &\mathbf{X}_{t_0-7*q+1}, \ldots, \mathbf{X}_{t_0-q+T_p})\end{aligned} \tag{14}$$

A spatio-temporal attention mechanism learns the correlations between the nodes. The spatial correlation matrix ($\mathbf{S} \in \mathbb{R}^{n\times n}$) is described in Equation (15), where the input of the spatial-temporal block is $\mathbf{X}_h = (\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_{T_r}) \in \mathbb{R}^{n\times d\times T_r}$, and T_r is the length of the temporal dimension. $\mathbf{V}_s$, $\mathbf{b}_s \in \mathbb{R}^{n\times n}$ are the weight matrix and the bias matrix, respectively, and $\boldsymbol{\theta}_1 \in \mathbb{R}^{T_r}$, $\boldsymbol{\theta}_2 \in \mathbb{R}^{T_r\times T_r}$, $\boldsymbol{\theta}_3 \in \mathbb{R}^{d}$ are the learnable parameters (Guo *et al.* 2019).

$$\mathbf{S} = \mathbf{V}_s \cdot \sigma((\mathbf{X}_h\boldsymbol{\theta}_1)\boldsymbol{\theta}_2(\boldsymbol{\theta}_3\mathbf{X}_h)^T + \mathbf{b}_s) \tag{15}$$

The temporal correlation matrix ($\mathbf{E} \in \mathbb{R}^{T_r\times T_r}$) is shown in Equation (16), where $\mathbf{V}_e$, $\mathbf{b}_e \in \mathbb{R}^{T_r-1\times T_r-1}$ are the weight matrix and the bias matrix, respectively, and $\boldsymbol{\theta}_{t,1} \in \mathbb{R}^{n}$, $\boldsymbol{\theta}_{t,2} \in \mathbb{R}^{d\times n}$, $\boldsymbol{\theta}_{t,3} \in \mathbb{R}^{d}$ are the learnable parameters (Guo *et al.* 2019).

$$\mathbf{E} = \mathbf{V}_e \cdot \sigma(((\mathbf{X}_h)^T\boldsymbol{\theta}_{t,1})\boldsymbol{\theta}_{t,2}(\boldsymbol{\theta}_{t,3}\mathbf{X}_h) + \mathbf{b}_e) \tag{16}$$

The graph convolution layer is described with the Chebyshev polynomial, and the integration of the three components ($\mathbf{H}_h$, $\mathbf{H}_d$, $\mathbf{H}_w$) into the output ($\mathbf{H}_t \in \mathbb{R}^{n\times T_p}$) is defined in Equation (17), where $\mathbf{W}_h \in \mathbb{R}^{n\times T_p}$, $\mathbf{W}_d \in \mathbb{R}^{n\times T_p}$ and $\mathbf{W}_w \in \mathbb{R}^{n\times T_p}$ are the learning parameters for each segment (Guo *et al.* 2019).

$$\mathbf{H}_t = \mathbf{W}_h \odot \mathbf{H}_h + \mathbf{W}_d \odot \mathbf{H}_d + \mathbf{W}_w \odot \mathbf{H}_w \tag{17}$$

3. CASE STUDY AND RESULTS

A case study is applied for the flow forecast with the six spatio-temporal graph neural network models and LSTM in the Upper Colorado River Basin, and the MSE, MAE, and NSE are used to compare the performance of the models. The objective of the forecast is to determine future flows given the flows previously observed from the measurement points in the river basin. The river basin is represented as an unweighted directed graph $G = (N, E)$, where N is the set of nodes, E is the set of edges. The observed features on the graph are $\mathbf{X} \in \mathbb{R}^{N\times d}$, where d is the number of features at each node. The measurement data are available as multivariate daily time series for all monitoring locations from the website of the US Geological Survey. The original sampling frequency is daily, with measurements taken at 11:59 p.m. on each date. The forecast lead time is one step ahead, corresponding to one day.

3.1. Study area

The Upper Colorado River Basin (UCRB) has a significant impact on the hydrology of the Colorado River, supports water storage in the area. Temperatures show a warming trend over the past century that starts the snowmelt earlier in spring, and the earlier snowmelt contributes more winter runoff and the reduction in summer inflows, which cause dryness in summer. The volume of the precipitation also changed, it increased in winter and decreased in summer. Due to changes in weather and decreased flow, the storage capacity of the reservoirs decreases (Dawadi & Ahmad 2012). The main reservoir in the UCRB is Lake Powell, and Figure 2 shows how the severe drought over the years has affected it. The figure in the upper left in 1999 shows the lake at a high level, but by 2021, after more than two decades of drought, water levels have decreased dramatically.

The variables measured for most of the reservoirs are inflow, total release, evaporation, bank storage, storage, unregulated inflow, pool elevation, power release, area, and bypass release. The inflow is determined from the gauged flow, which is the actual measured flow. Then the unregulated flow is determined as the flows as there was no upstream reservoir, evaporation, and bank storage. The flow is computed from the unregulated flows, as the water users are removed (Wheeler *et al.* 2019). The bank storage is the volume of water stored in the banks of the reservoir, which is determined from the difference in storage in the two timesteps multiplying by a bank storage coefficient. The evaporation is calculated from the evaporation rate, which corresponds to the current day of the year, and the surface area. In all reservoirs, the total release is the sum of the releases, which are the bypass release and the power release, which goes to a bypass tunnel and a hydroelectric power generation, respectively. The delta storage is determined from flow, total release, and evaporation in each reservoir, but it can also be computed from storage at each time step. The storage is determined from precipitation volume, evaporation, change in bank storage, and inflow (Bank Storage 2022). The rivers have one measured variable, which is the flow, which we call total release in this work similarly to the reservoirs, which increases every year during spring and summer and decreases in winter.

In this work, we forecast the total release of reservoirs and the flow of the rivers. Although meteorological variables are important drivers of inflow variability, we do not include them as input features currently. Instead, our aim is to assess the

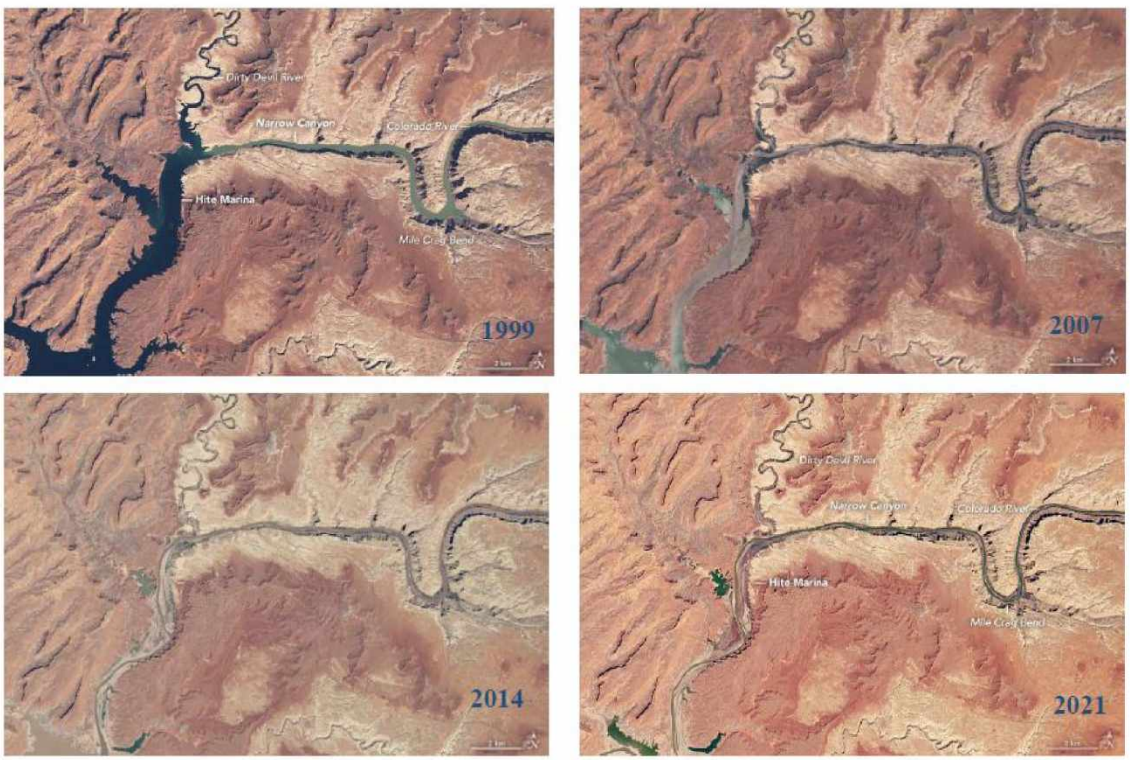

Figure 2 | The water level of Lake Powell over the years.

forecast capability of STGNNs when using only operationally available hydrologic variables that are consistently observed across all nodes. The measurement points and their names for the forecast are shown in Table 2, the forecast features are the flows and the total release of the rivers and reservoirs, respectively. The map of the river basin and the graph structure of the case study are shown in Figure 3, where 20 nodes are located as measuring points of the characteristics of rivers and reservoirs. The figure on the left shows the river basin and the measurement points, and the figure on the right shows the river basin as a directed, unweighted graph representing the flow directions. The adjacency matrix $A \in \{0, 1\}^{N \times N}$ is purely topological: $A_{ij} = 1$ if there is a direct connection from node i to node j, and $A_{ij} = 0$ otherwise.

3.2. Forecast with STGNN models

Six different graph neural networks and LSTM are used for the Upper Colorado River Basin to forecast the total releases of the reservoirs and the rivers. The data consist of a time series for 20 nodes up to five years (between 2019. 12. 13. and 2025. 05. 31.). We use the daily data available from the USGS, converted to SI units, without any further resampling or aggregation. The selected period contains no missing values in time series, so no gap filling or interpolation was required, and all extreme values were kept unchanged. The boxplot of the total releases of the reservoirs is shown in Figure 4(a), where the main plot illustrates the complete data range, and an inset plot shows a zoomed in view to better visualize the distribution for all reservoirs. Lake Powell (reservoir 7) exhibits higher total release values, as it is the main reservoir in the river basin, the median total release is $1, 3 \cdot 10^6 \text{ m}^3/\text{h}$. Taylor Park Reservoir, McPhee Reservoir, and Vallecito Reservoir (reservoir 5, 6, 9) have a smaller total release values than the others, since they are located upstream and receive less flow. The median of the other reservoirs is approximately $2 \cdot 10^5 \text{ m}^3/\text{h}$, and the data are similar to Crystal Reservoir, Morrow Point Reservoir, and Blue Mesa Reservoir (reservoir 2, 3, 4), because they are geographically close. The boxplot of the river flows is shown in Figure 4(b), where the main plot shows the complete data range, and the inset plot zooms in to better display the distribution for all river measurement points. Colorado River (measurement point 15) has the highest median flow, around $4 \cdot 10^5 \text{ m}^3/\text{h}$, as it is the main river in the basin. In contrast, Animas River (measurement points 16, 17, 18, 19) have lower flow values because they are located further upstream and have smaller contributing watersheds.

Table 2 | Number of rivers and reservoirs

Number	Name of measurement points
0.	Fontenelle Reservoir
1.	Flaming Gorge Reservoir
2.	Crystal Reservoir
3.	Morrow Point Reservoir
4.	Blue Mesa Reservoir
5.	Taylor Park Reservoir
6.	McPhee Reservoir
7.	Lake Powell
8.	Navajo Reservoir
9.	Vallecito Reservoir
10.	Yampa River at Deerlodge Park
11.	Yampa River near Maybell
12.	Gunnison River near Grand Junction
13.	Gunnison River at Delta
14.	Colorado River near Colorado-Utah state line
15.	Colorado River near Cisco
16.	Animas River at Farmington
17.	Animas River at Penny Lane near Flora Vista
18.	Animas River near Cedar Hill
19.	Animas River at Durango

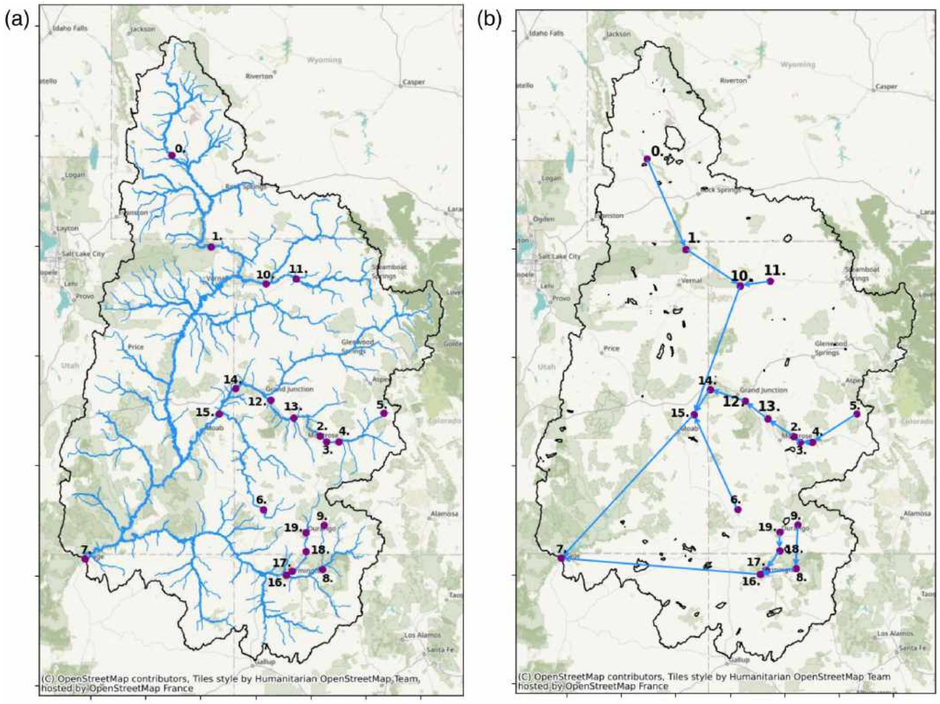

Figure 3 | Study area. (a) Map of the Upper Colorado River Basin and (b) Graph structure of the Upper Colorado River Basin.

Standardization is applied with StandardScaler, which scales the data with zero mean and 1 standard deviation. Then a moving window is applied to determine the target and feature arrays with 20 window length, where the first 20 elements of the time series go to the feature array, and the next go to the target array for each node. The feature array has a shape $1977 \times 20 \times 20$, while the target array has a shape $1977 \times 20 \times 1$. The input length of 20 days was chosen to match the typical hydrological characteristics of the river basin, but not so long that it increases the complexity of the model. Due to the high variability of meteorological conditions, we focus on short-term 1-day-ahead forecasting. Methodologically, the proposed spatio-temporal graph-based model could be extended to multi-step forecasting, for example designing the output layer to predict several future days at once.

A temporal graph is created with StaticGraphTemporalSignal from Pytorch Geometric Temporal with the feature and target matrix, and the edge list, and a train test split is made with the 0.9 train ratio over time. This split is chosen to ensure that the model is trained on a substantial portion of the data set to capture the temporal patterns and relationships effectively, since the data cover a period of five years.

To ensure a fair comparison between the different models, all of them were trained with a fixed and shared training setup which were manually tuned. All models utilize the Adam optimizer with a learning rate of 0.001, a batch size of 32, and are trained for 50 epochs without any early stopping criteria. None of the models implement dropout regularization, and all use MSE as their loss function. All seven models follow the same overall template: a single temporal or spatio-temporal processing block is applied first, then a ReLU nonlinearity, and finally a linear readout layer produces the forecast. The LSTM baseline uses a one-layer LSTM before the linear head. EvolveGCN-H and EvolveGCN-O each use one EvolveGCN recurrent layer, while GCRN and T-GCN both rely on a single GRU-style spatio-temporal recurrent cell, differing in how graph convolution is integrated into the recurrent update. STGCN uses one STConv spatio-temporal convolution block, and ASTGCN uses one attention-based spatio-temporal graph convolution block.

The forecast of the seven models is illustrated and compared in Appendix D for two main reservoirs and a main river from the 20 measurement points. The mean squared errors and the mean absolute errors in the train and the test set for all nodes in the river basin for the six STGNN models and the LSTM are shown in Figure 5. Overall, the T-GCN is the weakest and least stable model, exhibiting the highest median errors and the largest variability (wide interquartile range) in both

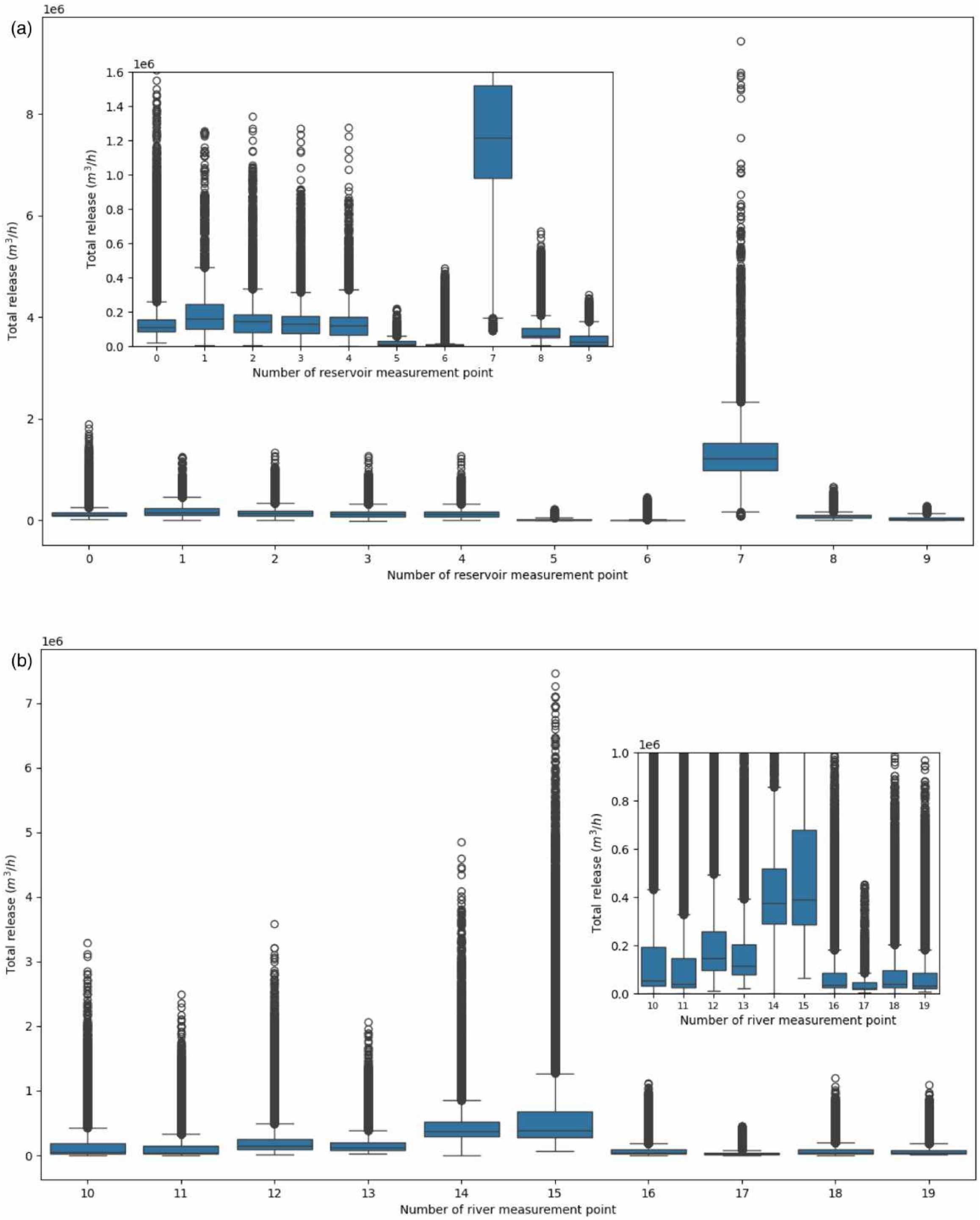

Figure 4 | Boxplot of the dataset. The box represents the interquartile range (25th–75th percentiles), and the dots indicate outliers. (a) Boxplot of total release across the reservoirs and (b) Boxplot of total release across the rivers.

MSE and MAE, especially in the test set. In the train set, GCRN and ASTGCN have the lowest errors, showing minimal variation. STGCN is generally competitive but displays some large outliers in the train MSE. In the test set, GCRN remains the most consistent, and STGCN and ASTGCN also show low variability, whereas the LSTM and the EvolveGCN variants have moderate performance with occasional outliers.

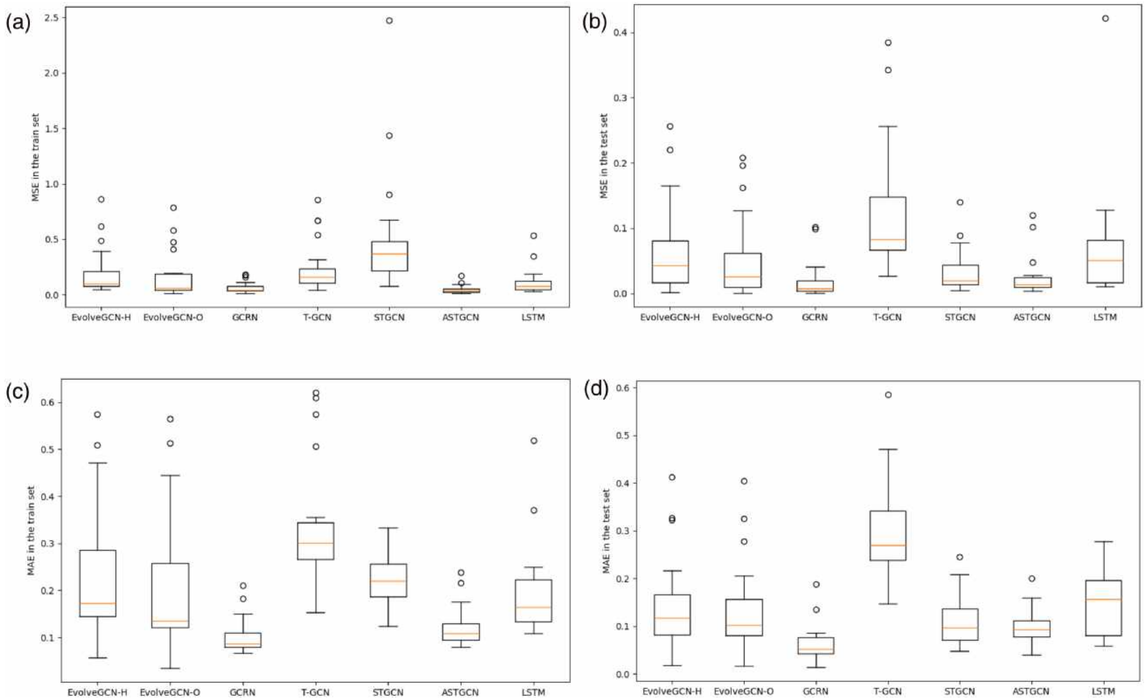

Figure 5 | Boxplots for the six STGNN models and LSTM. (a) Mean squared error in the train set using standardized data. (b) Mean squared error in the test set using standardized data. (c) Mean absolute error in the train set using standardized data and (d) Mean absolute error in the test set using standardized data.

The Nash–Sutcliffe efficiency (NSE) values for all nodes for the six STGNN models and the LSTM are shown in Appendix D for the train set in Table 4 and for the test set in Table 5. In the train set, all models achieve predominantly high NSE values, indicating a strong fit to the training data, with only a few more challenging nodes (Node 1, Node 7, Node 8). In the test set, the overall NSE decreases and the differences between models become more pronounced: GCRN and ASTGCN remain robust across most nodes, while T-GCN has strongly negative NSE at some nodes (Node 6, Node 7). The LSTM also shows node-specific failures in the test set (Node 0), while the EvolveGCN variants and STGCN generally provide moderate-to-strong performance with some variability across nodes.

To account for the performance variability of deep learning models, we repeated the training of each model two times using different random seeds. Table 3 shows the test-set MSE as mean ± standard deviation for three main nodes (Nodes 1, Node 7, and Node 14). Overall, GCRN and ASTGCN exhibit consistently low variability across runs, indicating stable optimization and robust performance. In contrast, T-GCN shows higher mean errors, suggesting less reliable convergence and greater sensitivity to initialization. LSTM displays increased variability at some nodes, whereas the EvolveGCN variants typically fall in between, with moderate performance.

The GCRN, STGCN, and ASTGCN models consistently achieve the lowest MSE and MAE values, making them the best-performing models. These models are able to capture the underlying temporal and spatial dependencies of the system effectively, which can be attributed to their architecture design. In contrast, the T-GCN model performs significantly worse than all others. This indicates that the simpler temporal GCN structure is not sufficient to represent the complexity of the hydrologic system.

Despite the strong forecasting capabilities, all models exhibit limitations in forecasting extreme events. Sharp peaks caused by snowmelt or heavy precipitation remain difficult to forecast, likely due to their rarity in the training data. This leads to a strong imbalance between normal and high-flow conditions, and with an MSE-based objective the models tend to prioritize the dominant low-to-moderate flows and smooth or under-estimate rare peaks. Performance may further degrade because the models rely mainly on discharge histories and do not explicitly include key drivers of rapid runoff generation (precipitation, temperature, or snow water equivalent). Incorporating such exogenous meteorological and snowpack inputs could improve peak timing and magnitude. From an operational perspective, missing large inflow/release peaks can be critical

Table 3 | MSE (mean ± standard deviation) computed in test set over two independent training runs with different random seeds for three main nodes

	EvolveGCN-H	EvolveGCN-O	GCRN	T-GCN	STGCN	ASTGCN	LSTM
(a) Node 1 (mean ± std over two runs).							
mean	0.1308	0.1434	0.0961	0.2072	0.1664	0.1217	0.3159
std	0.0031	0.0169	0.0029	0.0257	0.0779	0.0022	0.1059
(b) Node 7 (mean ± std over two runs).							
mean	0.1647	0.1647	0.0413	0.2468	0.0621	0.0579	0.1191
std	0.0003	0.0027	0.0060	0.0097	0.0156	0.0101	0.0090
(c) Node 14 (mean ± std over two runs).							
mean	0.0080	0.0066	0.0028	0.0603	0.0079	0.0065	0.0259
std	0.0015	0.0009	0.0002	0.0043	0.0016	0.0016	0.0092

for reservoir operations and flood management; therefore, the models are likely more reliable under typical conditions. We evaluated MSE values on upper quantiles of the standardized data for every model, which is 0.4634 for EvolveGCN-H, 0.3584 for EvolveGCN-O, 0.4273 for T-GCN, 0.1720 for GCRN, 0.4303 for STGCN, 0.1161 for ASTGCN, and 0.2889 for LSTM. The peak-flow errors remain substantially larger than the average-regime errors, with ASTGCN and GCRN showing the best robustness in high-flow conditions.

In general, GCRN, STGCN, and ASTGCN appear to be the most reliable models for minimizing MSE in water system forecasting. It can be concluded that, to compare the T-GCN and GCRN models, the Chebyshev graph convolution layer for the spatial layer performs better than the GCN layer.

4. CONCLUSION

River basin flow forecast has been investigated for the Upper Colorado River Basin with various spatio-temporal graph neural network models. These models utilize graph structures to capture spatial dependencies among different locations within the river basin while integrating temporal dynamics for improved forecast accuracy. The results indicate that graph neural network-based approaches are not only suitable for this specific basin but show promising potential as a generalizable tool for hydrological modeling in other river systems.

In the graph convolutional layers, three key matrices are considered: the degree matrix, the adjacency matrix, and the Laplacian matrix. These matrices facilitate the propagation of information between connected nodes. Specifically, in one approach, the sender node's features are multiplied by both the adjacency matrix and the degree matrix, ensuring the consideration of neighboring node influences while maintaining numerical stability. In another approach, the Chebyshev polynomial is applied to the sender node's features, allowing for a more flexible and higher-order representation of the graph structure. The other applied spatio layer is the attentional layer, where the features of the sender node are determined with an attention mechanism.

Six STGNNs are evaluated in this work. EvolveGCN and Temporal GCN models integrate the recurrent neural network model with graph convolution, with the latter allowing temporal variation in the convolutional weights. The Graph convolutional recurrent network has the Chebyshev polynomial as the graph convolutional layer, while the previous ones only determine with the adjacency matrix. In the Spatio-temporal graph convolutional network, two St-blocks are applied, and in the Attention-based spatio-temporal graph convolutional network an attention mechanism is applied.

The case study is in the Upper Colorado River Basin, with 20 nodes and 19 edges. The forecasts are made with the spatio-temporal graph neural network models for the flows of the river and the total releases of the reservoirs. The results show that the forecasts with the temporal graph neural network where the Chebyshev polynomial is used (GCRN) have the highest accuracy among the models. From an operational perspective, the proposed approach is computationally lightweight for this basin graph and could be deployed in a routine daily workflow. However, practical implementation would require procedures for periodic retraining, robustness to sensor data gaps, and monitoring performance under regime shifts such as multi-year droughts.

The future research direction contains the spatial distribution of the river basin, where forecasts can be made based on the input and output flows of the given area, allowing for improved accuracy in hydrological forecasting. Further exploration of hybrid models combining attention mechanisms with higher-order graph convolutions could also enhance the performance of the forecast. For further practical extension for reservoir operations, future work will also explore incorporating exogenous drivers (precipitation, temperature, snow information) and extending the setup from one-day-ahead to multi-step forecasting to better support operational decision-making.

ACKNOWLEDGEMENTS

The research presented in the article was carried out within the framework of the Széchenyi Plan Plus program with the support of the RRF-2.3.1-21-2022-00008 project. This research has been supported by Ministry of Culture and Innovation of Hungary from the National Research, Development and Innovation Fund through the Monitoring Complex Systems by Goal-Oriented Clustering Algorithms project under Grant OTKA 143482. Project no. 2024-1.1.1-KKV_FÓKUSZ-2024-00033 is co-funded by the NRDI Office and the National Innovation Agency with the support of the Hungarian State. This project was supported by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences (B0/00439/25/6).

DATA AVAILABILITY STATEMENT

All relevant data are included in the paper or its Supplementary Information. The developed code and all relevant data are available on the following link: https://github.com/DataCentricSE/STGNN.

CONFLICT OF INTEREST

The authors declare there is no conflict.

REFERENCES

Akkala, A., Boubrahimi, S. F., Hamdi, S. M., Hosseinzadeh, P. & Nassar, A. (2025) Spatio-temporal graph neural networks for streamflow prediction in the upper colorado basin, *Hydrology*, **12** (3), 60.

Bai, T. & Tahmasebi, P. (2023) Graph neural network for groundwater level forecasting, *Journal of Hydrology*, **616**, 128792.

Baker, S. A., Wood, A. W., Rajagopalan, B., Prairie, J., Jerla, C., Zagona, E., Butler, R. A. & Smith, R. (2022) The colorado river basin operational prediction testbed: a framework for evaluating streamflow forecasts and reservoir operations, *JAWRA Journal of the American Water Resources Association*, **58** (5), 690–708.

Bank Storage (2022) https://www.riverware.org/HelpSystem/8.5-Help/index.html#page/Objects/reservoirCommon.30.03.html# (Accessed: 24 June 2025).

Barker, L. J., Fry, M., Hannaford, J., Nash, G., Tanguy, M. & Swain, O. (2022) Dynamic high resolution hydrological status monitoring in real-time: the UK water resources portal, *Frontiers in Environmental Science*, **10**, 752201.

Dawadi, S. & Ahmad, S. (2012) Changing climatic conditions in the colorado river basin: implications for water resources management, *Journal of Hydrology*, **430**, 127–141.

Guo, S., Lin, Y., Feng, N., Song, C. & Wan, H. (2019) Attention Based Spatial-Temporal Graph Convolutional Networks for Traffic Flow Forecasting. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33, No. 01: AAAI Press, pp. 922–929.

Jin, G., Liang, Y., Fang, Y., Shao, Z., Huang, J., Zhang, J. & Zheng, Y. (2023) Spatio-temporal graph neural networks for predictive learning in urban computing: a survey, *IEEE Transactions on Knowledge and Data Engineering*, **36** (10), 5388–5408.

Liu, G., Ouyang, S., Qin, H., Liu, S., Shen, Q., Qu, Y., Zheng, Z., Sun, H. & Zhou, J. (2023) Assessing spatial connectivity effects on daily streamflow forecasting using Bayesian-based graph neural network, *Science of the Total Environment*, **855**, 158968.

Pareja, A., Domeniconi, G., Chen, J., Ma, T., Suzumura, T., Kanezashi, H., Kaler, T., Schardl, T. & Leiserson, C. (2020) EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34, No. 04: AAAI Press, pp. 5363–5370.

Seo, Y., Defferrard, M., Vandergheynst, P. & Bresson, X. (2018) Structured Sequence Modeling with Graph Convolutional Recurrent Networks. In: Cheng, L., Leung, A. C. S. & Ozawa, S. (eds.) *Neural Information Processing: 25th International Conference, ICONIP 2018, Siem Reap, Cambodia, December 13–16*, Part I. Lecture Notes in Computer Science, Vol. 11301. Cambodia: Springer, Cham, pp. 362–373.

Sha, J., Li, X., Zhang, M. & Wang, Z.-L. (2021) Comparison of forecasting models for real-time monitoring of water quality parameters based on hybrid deep learning neural networks, *Water*, **13** (11), 1547.

Wang, L., Jiang, Z., Song, L., Yu, X., Yuan, S. & Zhang, B. (2024) A groundwater level spatiotemporal prediction model based on graph convolutional networks with a long short-term memory, *Journal of Hydroinformatics*, **26** (11), 2962–2979.

Wheeler, K. G., Rosenberg, D. E. & Schmidt, J. C. (2019) Water resource modeling of the colorado river: present and future strategies, *White Pap*, **2**, Center for Colorado River Studies, Utah State University, Logan. pp. 47.

Yang, S., Zhang, Y. & Zhang, Z. (2023) Runoff prediction based on dynamic spatiotemporal graph neural network, *Water*, **15** (13), 2463.

Yu, B., Yin, H. & Zhu, Z. (2017) Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*.

Yu, J., Ma, T., Jia, L., Rong, H., Su, Y. & Wahab, M. M. A. (2024) Multivariate spatio-temporal modeling of drought prediction using graph neural network, *Journal of Hydroinformatics*, **26** (1), 107–124.

Zhao, L., Song, Y., Zhang, C., Liu, Y., Wang, P., Lin, T., Deng, M. & Li, H. (2019) T-gcn: a temporal graph convolutional network for traffic prediction, *IEEE Transactions on Intelligent Transportation Systems*, **21** (9), 3848–3858.

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C. & Sun, M. (2020) Graph neural networks: a review of methods and applications, *AI Open*, **1**, 57–81.

First received 17 July 2025; accepted in revised form 4 March 2026. Available online 7 May 2026