FIDLAR: Forecast-Informed Deep Learning Architecture for Flood Mitigation

Jimeng Shi[1], Zeda Yin[2], Arturo Leon[2], Jayantha Obeysekera[2], Giri Narasimhan[1]

[1]Knight Foundation School of Computing and Information Sciences, Florida International University
[2]Department of Civil and Environmental Engineering, Florida International University
[3]Sea Level Solutions Center, Florida International University
{jshi008, zyin005, arleon, jobeysek, giri}@fiu.edu

Abstract

In coastal river systems, floods, often during major storms or king tides, severely threaten lives and property. However, hydraulic structures such as dams, gates, pumps, and reservoirs exist in these river systems, and these floods can be mitigated or even prevented by strategically releasing water before extreme weather events. A standard approach used by local water management agencies is the "rule-based" method, which specifies predetermined water prereleases based on historical human experience, but which tends to result in excessive or inadequate water release. Iterative optimization methods that rely on detailed physics-based models for prediction are an alternative approach. Whereas, such methods tend to be computationally intensive, requiring hours or even days to solve the problem optimally. In this paper, we propose a **F**orecast **I**nformed **D**eep **L**earning **Ar**chitecture, FIDLAR, to achieve rapid and near-optimal flood management with precise water prereleases. FIDLAR seamlessly integrates two neural network modules: one called the `Flood Manager`, which is responsible for generating water pre-release schedules, and another called the `Flood Evaluator`, which evaluates those generated schedules. The Evaluator module is pre-trained separately, and its gradient-based feedback is utilized to train the Manager model, ensuring near-optimal water pre-releases. We have conducted experiments with a flood-prone coastal area in South Florida. Results show that FIDLAR is several orders of magnitude faster than currently used physics-based approaches while outperforming baseline methods with improved water pre-release schedules.

1 Introduction

Floods can result in catastrophic consequences with considerable loss of life (Jonkman and Vrijling 2008), huge socio-economic impact (Wu et al. 2021), property damage (Brody et al. 2007), and environmental devastation (Yin et al. 2023a). They pose a threat to food and water security as well as sustainable development (Kabir and Hossen 2019). What is even more alarming is that research indicates global climate change may lead to a drastic increase in flood risks in terms of both frequency and scale (Wing et al. 2022), e.g., coastal flood risks due to sea-level rise (Sadler et al. 2020). Thus, effective flood management is of utmost importance.

To mitigate flood risks, water management agencies have built controllable hydraulic structures such as dams, gates, pumps, and reservoirs in river systems (Kerkez et al. 2016). However, determining the optimal *control schedules* of these hydraulic structures is a challenging problem (Bowes et al. 2021). Rule-based methods (Sadler et al. 2019) formulate control schedules based on insights gained from historically observed data. The rules represent the collective wisdom gathered over decades of experience in managing specific river systems. Nevertheless, these rules may expose vulnerabilities while dealing with extremely rare events and may not offer effective solutions for complex river systems under such conditions (Schwanenberg, Becker, and Xu 2015).

The flood mitigation problem is usually treated as an optimization task (Karimanzira 2016), where variables such as control schedules for hydraulic structures are optimized to achieve the desired water levels to effectively mitigate flood risks (Zarei et al. 2021). Random initialization followed by soft-computing techniques such as genetic algorithm (GA) (Leon et al. 2020) is often employed to perform the optimizations. Subsequently, physics-based simulator models such as HEC-RAS and SWMM are used to assess generated control schedules (Sadler et al. 2019; Leon et al. 2014; Yin et al. 2023b). However, such methods are prohibitively slow since they require thousands of time-consuming physics-based simulations (Jafarzadegan et al. 2023).

Over the past decade, machine learning (ML) has become an increasingly powerful tool in various aspects of flood management, including flood prediction (Shi et al. 2023c), flood detection (Tanim et al. 2022), susceptibility assessment (Saha et al. 2021), and post-flood management (Munawar et al. 2019). Despite these advances, ML-based methods have yet to be widely applied to flood mitigation tasks. We address this gap by introducing a fully machine learning-based framework, comprising a *generator* to generate control schedules and a *simulator* to assess them. The ML-based framework offers significantly faster response times than compute-intensive physics-based approaches, enabling real-time flood management. Furthermore, the proposed ML-based method allows the simulator to actively guide the generator through gradient descent, thus seamlessly integrating the generator and simulator components and leveraging the differentiability of the learned simulator model (Jyothir et al. 2023; LeCun 2022; Bharadhwaj, Xie, and Shkurti 2020).

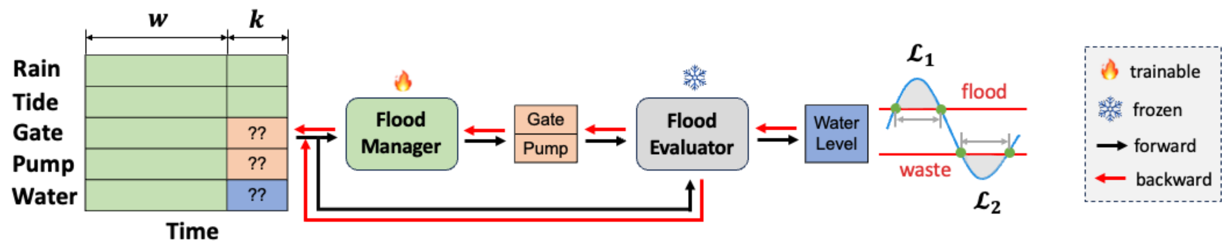

Figure 1: Forecast-Informed Deep Learning Architecture (FIDLAR). Input data consists of five categories of variables as shown in the left table. The variables w and k are the lengths of the past and prediction windows, respectively. The parts colored green are provided as inputs, while the orange and blue parts (with question marks) are outputs. The `Flood Manager` and `Flood Evaluator` represent deep learning (DL) models, the former to predict control schedules of controllable hydraulic structures (e.g., gates and pumps) to pre-release water, and the latter to predict the resulting water levels for those control schedules. Loss functions, $\mathcal{L}_1$ and $\mathcal{L}_2$, penalize the *flooding* and *water wastage* beyond pre-specified thresholds, respectively.

To this end, we propose **F**orecast **I**nformed **D**eep **L**earning **Ar**chitecture, FIDLAR, for flood management. FIDLAR's characteristics are summarized as follows:

- FIDLAR seamlessly combines two DL models in series: `Flood Manager` and `Flood Evaluator`. The former model is responsible for generating water pre-release schedules, while the latter model accurately forecasts the resulting water levels. Moreover, with the gradient-based planning, and differentiability of trained `Evaluator`, it can reinforce the `Manager` to generate better schedules.
- FIDLAR is a data-driven approach, learning flood mitigation strategies from historically observed data. Once trained, it offers rapid response capabilities, highlighting the advantages of DL-based models over physics-based models, particularly for real-time flood management.
- FIDLAR is a model-agnostic framework, where both the `Manager` and `Evaluator` could be any type of DL model trained with differentiable loss functions that allow back-propagation.
- FIDLAR is trained with a customized loss function to balance flood risks and water wastage at the same time, which is a multitask learning method.

2 Related Work

Flood Prediction. Physics-based mechanic models (e.g., HEC-RAS, SWMM) have been widely used to simulate water levels and flows in river systems (Peker et al. 2024; Rahman and Ali 2024; Gomes Jr et al. 2023; Rivett et al. 2022). However, these models are computationally inefficient and fall short of capturing precise knowledge of study domains (Bentivoglio et al. 2022). Therefore, diverse machine learning (ML) and deep learning (DL) models have been studied as surrogates to simulate water levels and flows. For example, support vector machines (SVMs) have been used to predict the urban flash floods (Yan et al. 2018; Choubin et al. 2019), multivariate regression models were adopted to estimate flood volumes and peak flows (Yang and Chang 2020), random forests and K-nearest neighbors were explored for urban flood inundation mapping (Castro-Gama et al. 2014), gaussian process learning models for fast and accurate flood inundation simulation (Fraehr et al. 2023). Furthermore, deep learning models, such as recurrent neural networks, convolutional neural networks, and Transformers, have been employed for flood inundation (Zhou et al. 2021) and flood prediction (Shi et al. 2023b,c).

Flood Mitigation. Flood mitigation seeks control schedules of those hydraulic structures to avoid or mitigate flood risks, which requires flood prediction models as simulators to evaluate the control schedules. Researchers have attempted to leverage the genetic algorithm and the pattern search to generate control schedules and physics-based models (e.g., EPA-SWMM5, HEC-RAS) as water simulators (Sadler et al. 2019; Leon et al. 2014, 2020). The Lake Mendocino Operations (LMO) model was developed to simulate operations of Lake Mendocino such as release constraints for flood control and water supply operations (Delaney et al. 2020). However, such methods are computationally intensive since they require thousands of time-consuming simulation trials with physics-based models (Jafarzadegan et al. 2023). Additionally, the genetic algorithm and pattern search techniques are usually heuristic, without positive feedback or guidance to better control those hydraulic structures.

3 Problem Formulation

Flood management or mitigation aims to manage water levels before extreme weather events. It involves predicting control schedules for hydraulic structures such as gates and pumps within the river system, denoted as $X^{gate,pump}_{t+1:t+k}$, spanning k time points into the future from $t+1$ to $t+k$. This prediction takes as input historical data on all possible variables (see Figure 5), X, from the preceding w time points, in conjunction with reliably forecasted covariates (such as rainfall and tide) for the next k time points. Then we could train a deep learning (DL) model, $\mathcal{M}_\theta$, with parameters θ:

$$\mathcal{M}_{\theta_M} : (X^{all}_{t-w+1:t}, X^{cov}_{t+1:t+k}) \rightarrow X^{gate,pump}_{t+1:t+k}, \quad (1)$$

where the subscripts represent the time ranges, and the superscripts refer to the variables under consideration. Superscripts are dropped when all variables are taken into account. The superscript cov refers specifically to the reliably predicted covariates (e.g., rain, tides).

4 Methodology

4.1 Overview

Intuitively, an ML model can be trained to learn the function $\mathcal{M}_{\theta_M}$. However, a key challenge lies in the historical data, which often reflects control schedules that led to flooding or other suboptimal outcomes, making it unsuitable as ground-truth data for traditional supervised learning. To overcome this, we train the model, $\mathcal{M}_{\theta_M}$, by leveraging the differentiability of a learned simulator model, $\mathcal{E}_{\theta_E}$. First, we pre-train an independent and accurate `Flood Evaluator` using extensive historical data to model the *consequences* (e.g., water levels) of various *actions* (e.g., control schedules). We then frame the control schedule planning in `Flood Manager` as an optimization problem, seeking actions that minimize undesirable outcomes - floods or water wastage. Both the `Evaluator` and `Manager` are implemented as neural networks, with the framework illustrated in Figure 1.

4.2 Flood Evaluator

`Flood Evaluator` is tasked with accurately forecasting water levels at designated points of interest within river systems for any control schedule of gates and pumps. The underlying transfer function of the `Evaluator` is:

$$\mathcal{E}_{\theta_E} : (X^{all}_{t-w+1:t}, X^{cov}_{t+1:t+k}, X^{gate,pump}_{t+1:t+k}) \to X^{water}_{t+1:t+k}. \tag{2}$$

The `Evaluator` is trained independently using large-scale historical data to achieve highly accurate water level predictions for any given set of conditions and control schedules. Therefore, once the `Evaluator` is well trained, its parameters are frozen while training the `Manager`, where it plays the role of a trained "referee" - scoring control schedules generated by the `Manager` by predicting the resulting water levels. It also serves to backpropagate the gradient descent feedback, guiding the `Manager` to produce more effective control schedules of gates and pumps.

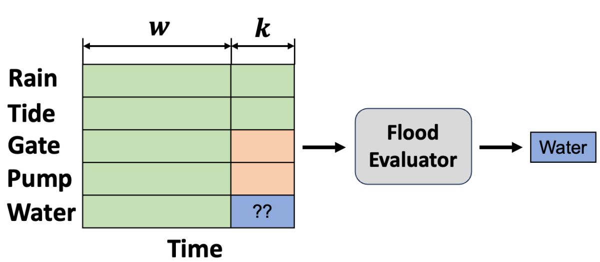

Figure 2: **Flood Evaluator.** The parts shaded green are used as inputs. Water levels (blue) are the outputs.

4.3 Flood Manager

`Flood Manager` is to produce control schedules for hydraulic structures (i.e., gates and pumps), taking as inputs reliably predictable future information (rain, tide) and all historical data. Since no ground truth is available, it is trained with the differentiability of the learned `Evaluator` model. Therefore, we connect the `Manager` with the `Evaluator`

Algorithm 1: Training algorithm of FIDLAR

Input: recent past data: $\mathbf{X}^{all}_{t-w+1,t}$
near future data: $\mathbf{X}^{cov}_{t+1,t+k} = \mathbf{X}^{rain,tide,gate,pump}_{t+1,t+k}$
Parameter: θ_E, θ_M: parameters of Evaluator and Manager; w, k : length of past and prediction windows

```
// Train Flood Evaluator, ℰ_θE
initialize learnable parameters θ_E
for i = 1, ..., N epochs do
    MiniBatch ← ({X^all_{t−w+1,t}, X^cov_{t+1,t+k}}, X^water_{t+1,t+k})
    X̂^water_{t+1,t+k} ← ℰ_θE(X^all_{t−w+1,t}, X^cov_{t+1,t+k})
    ℒ_E ← (1/k) Σ_{j=1}^{k} ||X̂^water_j − X^water_j||^2
    ∇_θE ← BackwardAD(ℒ_E)
    θ_E ← θ_E − η∇_θE
end for
return trained Flood Evaluator, ℰ_θE
// Train Flood Manager, ℳ_θM, with frozen ℰ_θE
initialize learnable parameters θ_M
while X^water_{t,t+k} violates either threshold do
    MiniBatch ← ({X^all_{t−w+1,t}, X^rain,tide_{t+1,t+k}}, X^gate,pump_{t+1,t+k})
    X̂^gate,pump_{t+1,t+k} ← ℳ_θM(X^all_{t−w+1,t}, X^rain,tide_{t+1,t+k})
    X̂^water_{t+1,t+k} ← ℰ_θE(X^all_{t−w+1,t}, X^rain,tide_{t+1,t+k}, X̂^gate,pump_{t+1,t+k})
    ℒ_E = c_1 · ℒ_1(X̂^water_{t+1,t+k}) + c_2 · ℒ_2(X̂^water_{t+1,t+k})
    ∇_θM ← BackwardAD(ℒ_E)
    θ_M ← θ_M − η∇_θM
end while
return trained Flood Manager, ℳ_θM
```

where the output of Eq. (1) is injected into Eq. (2):

$$\mathcal{E}_{\theta_E}(X^{all}, X^{cov}_{t+1:t+k}, \mathcal{M}_{\theta_M}(X^{all}, X^{cov}_{t+1:t+k})) \to X^{water}_{t+1:t+k}, \tag{3}$$

where $X^{all} = X^{all}_{t-w+1:t}$ and θ_M and θ_E are the parameters of `Manager` and `Evaluator`.

The resulting output of water levels can be used to compute the loss in Eq. (6), representing evaluation scores for generated control schedules. Gradient descent (Ruder 2016) can be back-propagated as the feedback to update the parameters of the `Manager`. The parameter update is presented:

$$\theta_M := \theta_M - \alpha \cdot \frac{\partial \mathcal{L}}{\partial \theta_M}, \tag{4}$$

where α is the learning rate and $\frac{\partial \mathcal{L}}{\partial \theta_M}$ is the partial derivative of the compound function $\frac{\partial \mathcal{L}}{\partial \theta_M} = \frac{\partial \mathcal{L}}{\partial \mathcal{E}} \cdot \frac{\partial \mathcal{E}}{\partial \mathcal{M}} \cdot \frac{\partial \mathcal{M}}{\partial \theta_M}$. The training details of FIDLAR are given in Algorithm 1.

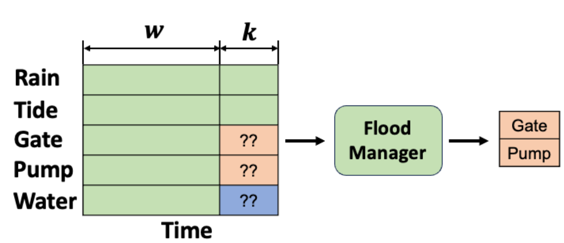

Figure 3: **Flood Manager.** The parts shaded green (historical data) are the inputs, and the parts shaded orange are the outputs. The water levels shaded blue are not predicted.

4.4 Custom Loss Function

Loss functions are critical in steering the learning process. An obvious one is the total time (Figure 4a) for which the water levels either exceed the *flooding threshold* or dip below the *water wastage threshold*. Another related metric is the extent to which the limits are exceeded to signify the severity of floods or water wastage (Figure 4b). The lower threshold for flood management is important in practice, since it prevents water wastage, thereby supporting irrigation, facilitating navigation, and maintaining ecological balance. It also prevents the optimization methods from trivially recommending the depletion of valuable water resources to prevent future flooding. The $\mathcal{L}_1$ and $\mathcal{L}_2$ represent the *flooding* and *water wastage* losses, respectively, and the final loss function is a balanced combination as shown in Eq. (6).

$$\begin{aligned} \mathcal{L}_1 &= \sum_{i=1}^{N} \sum_{j=t+1}^{t+k} \| max\{\hat{X}_{i,j}^{water} - X_i^{flood}, 0\} \|^2, \\ \mathcal{L}_2 &= \sum_{i=1}^{N} \sum_{j=t+1}^{t+k} \| min\{\hat{X}_{i,j}^{water} - X_i^{waste}, 0\} \|^2, \end{aligned} \tag{5}$$

where N is the number of water level locations of interest; k is the length of prediction horizon; X^{flood} and X^{waste} represent the thresholds for flooding and water wastage; and the capped version, $\hat{X}^{water}$, is obtained using the `Evaluator` module. The combined loss function is given by:

$$\mathcal{L}_{total} = c_1 \cdot \mathcal{L}_1 + c_2 \cdot \mathcal{L}_2, \tag{6}$$

where c_1/c_2 dictates the relative importance of $\mathcal{L}_1$ and $\mathcal{L}_2$.

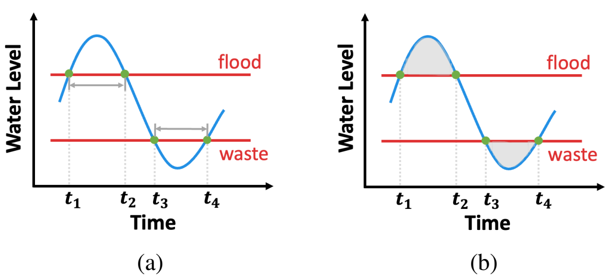

Figure 4: The two red bars represent a threshold of flooding and a threshold of water wastage. Shown are (a) the time spans when these thresholds are crossed, and (b) the areas between water level curves and threshold bars. Violations of the upper and lower thresholds are captured in $\mathcal{L}_1$ and $\mathcal{L}_2$.

5 Graph Transformer Network

The `Manager` and `Evaluator` modules described so far are model agnostic. We tried many existing architectures for them, as discussed in Section 6.2. We devise the Graph Transformer Network (GTN) architecture by combining graph neural networks (GNNs), attention-based transformer networks, long short-term memory networks (LSTMs), and convolutional neural networks (CNNs). GNN and LSTM modules are combined to learn the spatiotemporal dynamics of water levels, while the Transformer and CNN modules focus on extracting feature representations from the covariates. The *attention* mechanism (Vaswani et al. 2017) is used to discern interactions between covariates and water levels, as shown in Eq. (7) below. Figure 6 presents the GTN architecture, which is used for both `Evaluator` and `Manager`, but with minor changes accordingly of the inputs and outputs (see Figures 2 and 3). We have two GNN layers with 32 and 16 channels, one LSTM layer with 16 units, one CNN with 96 filters, and one Transformer encoder with 3 heads.

$$\begin{aligned} Atte(Q,K,V) &= softmax(\frac{Q^{cov}(K^{water})^T}{\sqrt{d}})V^{water} \\ &= softmax(\frac{Q^{water}(K^{cov})^T}{\sqrt{d}})V^{cov}, \end{aligned} \tag{7}$$

where T denotes the transpose operation; $water$ and cov represent water levels and covariates; and d is the length of projection embedding where $d = d_q = d_k = d_v = 128$.

6 Experiments

6.1 Dataset

We obtained data from the South Florida Water Management District's (SFWMD) DBHydro database (District 2023) for the coastal stretch in South Florida. The data set consists of hourly observations for water levels and external covariates from January 1, 2010 to December 31, 2020. As shown in Figure 5, the river system has two branches and includes several hydraulic structures (gates, pumps) to control water flows. We aim to predict effective control schedules on hydraulic structures (gates, pumps) to minimize flood risks at four specific locations marked by green circles.

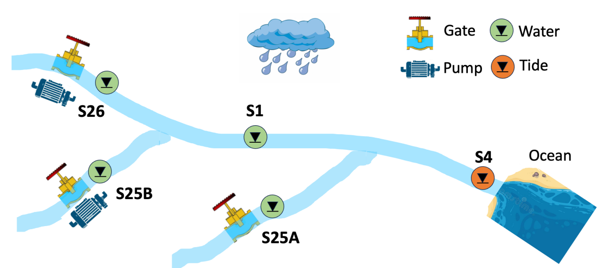

Figure 5: Schemetic diagram of study domain. There are three water stations with hydraulic structures (S26, S25B, S25A), one simple water station, S1 (green circle in the middle), and a station monitoring the tide level from the ocean.

Feature	Interval	Unit	#Var.	Location
Rainfall	Hourly	$inch/h$	1	-
Tide	Hourly	ft	1	S4
Pump	Hourly	ft^3/s	2	S25B, S26
Gate	Hourly	ft	3	S25A, S25B, S26
Water	Hourly	ft	4	S25A, S25B, S26, S1

Table 1: Summary of the data set.

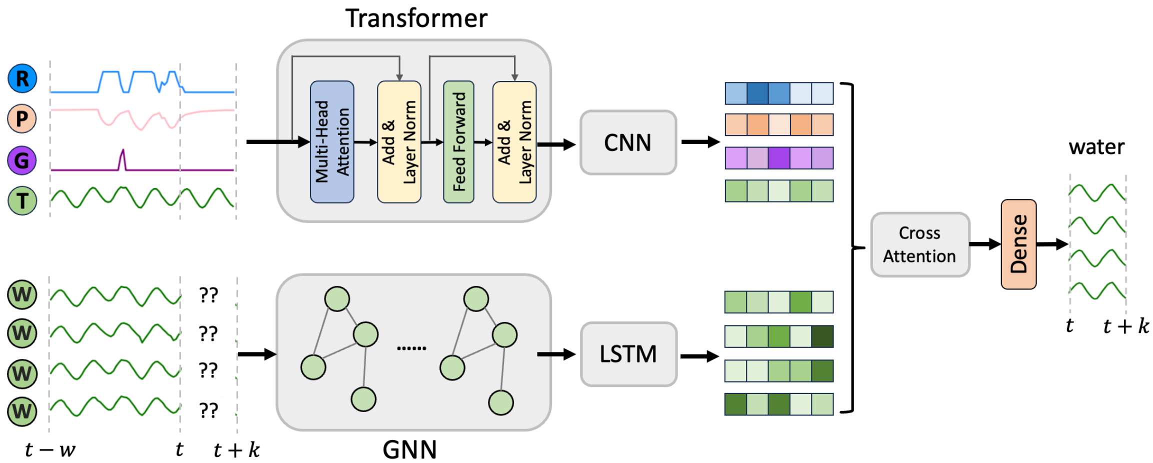

Figure 6: Graph Transformer Network for `Flood Evaluator`. Input variables include Rainfall, Pump, Gate, Tide, and Water levels as shown in Figure 5 and Table 1, which are generically denoted by R, T, G, P, and W. Output is water levels.

6.2 Experimental Design

The sliding input window (Li et al. 2014) (also known as look-back window (Gidea and Katz 2018) strategy was used to process the entire dataset (Shi et al. 2023a)). For consistency, we used a look-back window of length $w = 72$ hours and a prediction window of length $k = 24$ hours. The dataset was split in chronological order with the first 80% for training and the remaining 20% for testing. The eight DL methods below are used for `Flood Manager` and `Flood Evaluator`. We run all experiments on one NVIDIA A100 GPU with 80GB memory.

- **MLP** (Suykens, Vandewalle, and De Moor 1995): Multilayer perceptron can learn non-linear dependencies;
- **RNN** (Medsker and Jain 2001): Recurrent neural networks are good at processing sequential data;
- **CNN** (O'Shea and Nash 2015): A 1D convolutional neural network;
- **GNN** (Kipf and Welling 2016): Graph neural network with nodes representing variables and edges representing spatial dependencies;
- **TCN** (Bai, Kolter, and Koltun 2018): Temporal dilated convolutional network with an exponentially large receptive field;
- **RCNN** (Zhang and Dong 2020): Combined RNN and CNN model for time series forecasting;
- **Transformer** (Vaswani et al. 2017): Attention-based network for sequence modeling (only encoder);
- **GTN** (Ours): Combining GNNs with LSTMs, CNNs, and transformers, as described in Figure 6.

Flood Prediction. The role of the `Flood Evaluator` is to forecast flood events by predicting water levels for given input conditions. We set the upper threshold (flood level) at 3.5 feet and the lower threshold (wastage level) at 0.0 feet. However, the methods remain consistent for many reasonable choices of threshold values. We measured accuracy using multiple metrics: (a) mean absolute error (MAE), (b) root mean squared error (RMSE) computed between the predicted and actual water levels, (c) number of time points where the upper or lower thresholds are breached, and (d) the area between water level curves and threshold bars. Table 2 demonstrates that our model `GTN` outperforms other models with predictions (in red) most closely aligned with the ground truth (in blue) while achieving the lowest MAE and RMSE (in orange). Therefore, we choose our GTN model as `Evaluator` while training `Manager` in FIDLAR. We provide more results in the appendix of our arXiv version[1].

Flood Mitigation with FIDLAR. FIDLAR requires both `Evaluator` and `Manager` components. For the `Manager` model, we experimented with one rule-based method, and two genetic algorithms – one with a physics-based HEC-RAS evaluator (Leon et al. 2020) and one with our DL-based GTN evaluator, and several DL-based managers using MLP, RNN, CNN, GNN, TCN, RCNN, Transformer, and GTN. FIDLAR was measured using (a) the number of time steps where the upper/lower thresholds are exceeded for the water levels, and (b) the area between the water level curves and the threshold bars. Table 3 shows that all DL-based methods consistently performed better for site S1 than rule-based and GA-based approaches. Furthermore, GTN has the best performance under all four metrics, whether it is to control floods or water wastage. More results are in the appendix of our arXiv version[1].

We visualize water levels for a short period spanning 18 hours from September 3rd (09:00) to September 4th (03:00) in 2019 for the S1 location. Figure 7 indicates that FIDLAR equipped with GTN model (purple curve) has led to water levels within the upper and lower thresholds, satisfying pre-defined requirements. Moreover, FIDLAR presents the best control (i.e., the least water levels beyond thresholds) for flood mitigation and water waste compared to other baselines. We zoomed in on a 2.5-hour period of decreased water levels with predicted gate and pump schedules.

[1]Link: *https://arxiv.org/abs/2402.13371*

Methods	MAE (ft)	RMSE (ft)	Over Timesteps	Over Area	Under Timesteps	Under Area
Ground-truth	-	-	96	14.82	1,346	385.80
HEC-RAS	0.174	0.222	68	10.07	1,133	325.33
MLP	0.065	0.086	147	27.96	1,677	500.41
RNN	0.054	0.072	110	17.12	1,527	441.41
CNN	0.079	0.104	58	5.91	1,491	413.22
GNN	0.054	0.070	102	15.90	1,569	462.63
TCN	0.050	0.065	47	5.14	1,607	453.63
RCNN	0.092	0.110	37	4.61	1,829	553.20
Transformer	0.050	0.066	151	25.95	1,513	434.13
GTN (ours)	0.040	0.056	100	15.64	1,390	398.84

Table 2: Comparison of model performances for the `Flood Evaluator` on the test set, specifically at time t+1 for measurement station S1. The terms "Over Timesteps" and "Under Timesteps" indicate the number of time steps during which water levels exceed the upper threshold or fall below the lower threshold, respectively. Similarly, "Over Area" and "Under Area" pertain to the area between the water level curve and upper or lower threshold, as was illustrated in Figure 4. Results in orange are the lowest in that column while results in red are the closest to the ground truth (in blue).

Method	Manager	Over Timesteps	Over Area	Under Timesteps	Under Area
Rule-based		96	14.82	1,346	385.8
GA-based	Genetic Algorithm*	-	-	-	-
	Genetic Algorithm†	86	16.54	454	104
DL-based	MLP	91	13.31	1,071	268.35
	RNN	35	3.97	351	61.05
	CNN	81	11.22	1,163	314.37
	GNN	31	3.72	429	84.31
	TCN	39	3.77	306	55.12
	RCNN	29	3.28	328	58.68
	Transformer	85	11.54	1,180	310.16
	GTN (Ours)	**22**	**2.23**	**299**	**53.34**

Table 3: Comparison of model performances for the `Flood Manager` on the test set, specifically at time t+1 for measurement station S1. The * denotes that the GA method was used with a physics-based (HEC-RAS) evaluator. The $-$ denotes that the experiments were timed out. The † denotes the GA method was used with the GTN as the evaluator. All other rows are DL-based flood managers with a DL-based GTN as the evaluator. Results in bold are the best in that column.

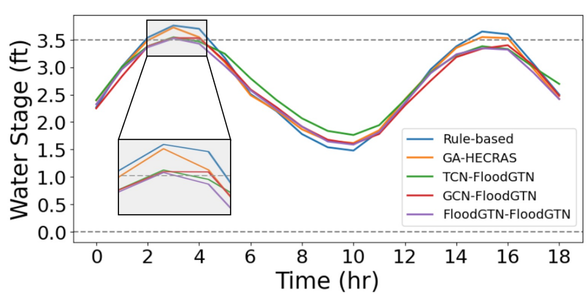

Figure 7: Visualization of water levels with various methods for flood mitigation. We zoomed in $t = 2 \sim 4.5$ in gray. 'A'–'B' in legend represents the `Manager` and `Evaluator`.

6.3 Ablation Study

The *ablation* study in Table 4 quantifies the contribution of each component of GTN, by measuring the performance of GTN after removing each of the individual components.

6.4 Analysis of Computational Time

Since FIDLAR was designed for real-time flood control, we measured the running times of the models used in this work. Table 5 shows the running times for the whole flood prediction and mitigation system in its training and test phase. All the DL-based approaches in the test phase are several orders of magnitude faster than the currently used physics-based and GA-based approaches for the flood mitigation task. Rapid inference is a critical property of data-driven DL methods. The table also shows the training times for the DL-based approaches, although they are not necessary for the deployment in reality.

6.5 Explainability

Attention-based methods allow us to calculate the "attention scores" assigned to an input variable to compute a specific output variable. This is exemplified in the heatmap shown in Figure 8, which shows the attention scores assigned to the tide (columns) to compute the gate schedule output (rows) for 24 hours into the future. Note that there are 96 columns and 24 rows because we use 72 hours of past tidal observations and 24 hours of future predicted tidal data to predict 24 hours of the gate schedule into the future. The rows $[0, 23]$ correspond to the 24 hours into the future, while the columns $[0, 95]$ also include 72 hours of the recent past and 24 hours of the future predicted tidal data. Therefore, $t = 72$ corresponds to the "current" time point, and the columns $[72, 95]$ correspond to the same time points as rows $[0, 23]$.

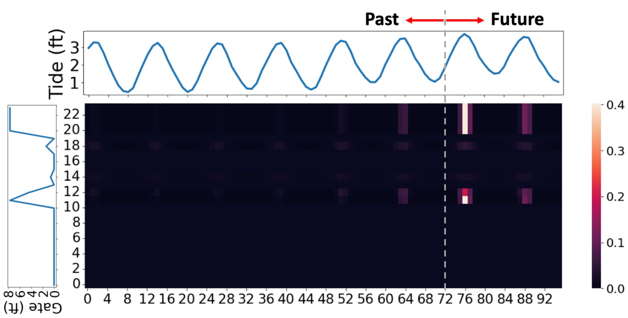

Figure 8: Importance scores of tide input. x and y axes are the tide and gate over time.

Method	Over Timesteps	Over Area	Under Timesteps	Under Area
w/o CNN	37	4.37	476	85.54
w/o Transformer	32	3.57	325	57.42
w/o GNN	56	5.90	479	86.22
w/o LSTM	35	4.34	329	56.74
w/o Attention	32	3.59	341	60.48
GTN (Ours)	**22**	**2.23**	**299**	**53.34**

Table 4: Ablation study for flood mitigation for the entire test set (for time point t+1 at S1). The last row indicates the performance of the FIDLAR system with GTN as proposed in Figure 6. The best results in the last row are in bold.

7 Discussion

Interpreting Attention Map. The explainability feature, which is shown with an example in Figure 8, can provide significant insights into our results. Firstly, we point out that the brightest patches are in the last 24 columns of the heatmap. Thus FIDLAR pays greater attention to the 24 hours of future predicted tidal data than the past 72 hours, highlighting the importance of forecast information to a DL-based approach to flood mitigation. While tides may have a more predictable pattern over time, the contribution of rain on the water levels can also be seen for other time points. A second critical insight is that the brightest attention patches are in columns where the tide is at its highest is critical to the prediction of gate schedules. Additionally, the water level at the first high tide peak after the "current" time is more significant than the other two. Third, the gate schedule has peaks at times $t = 11$ hour and $t = 22$ hour into the future, which correspond to the lowest points of the tide. This implies that the optimal time for pre-releasing water is during low tide phases. Opening gates during high tide periods in coastal river systems is less advisable, as it may lead to water flowing back upstream from the ocean. Finally, we observe that there is a light patch around column $t = 65$ hour, suggesting mild attention for the previous high tide peak, but almost no attention to any of the peaks before that. This again suggests that we could have chosen to use a smaller window for the past input. Doing this analysis could provide evidence for the right value of w, the size of the look-back window.

Model	Flood Prediction Train	Flood Prediction Test	Flood Mitigation Train	Flood Mitigation Test
HEC-RAS	-	45 min	-	-
Rule-based	-	-	-	-
GA*	-	-	-	-
GA†	-	-	-	est. 30 h
MLP	35 min	1.88 s	58 min	6.13 s
RNN	243 min	8.57 s	54 min	12.75 s
CNN	37 min	1.93 s	17 min	5.84 s
GNN	64 min	3.13 s	29 min	7.26 s
TCN	60 min	4.57 s	45 min	9.06 s
RCNN	136 min	8.61 s	61 min	13.27 s
Transformer	43 min	2.38 s	23 min	6.76 s
GTN (Ours)	119 min	2.95 s	35 min	4.90 s

Table 5: Running time for flood prediction and mitigation. The running time for the rule-based method is not reported since historical data was directly used. GA*, which combines a GA-based tool and HEC-RAS, took too long and was not reported, although the time for a small data set is reported in the appendix of our arXiv version. GA†, which combines the GA-based tool with GTN, also took too long but was estimated using a smaller sample.

8 Conclusions

In this work, we present the shortcomings of the current approaches for flood mitigation and propose FIDLAR, a DL-based tool to address the problem. FIDLAR can compute water "pre-release" schedules for hydraulic structures in a river system to achieve effective and efficient flood mitigation, while ensuring that water wastage is avoided. This was made possible by the use of well-crafted loss functions for the DL models. The dual component design (with a `Manager` and an `Evaluator`) is a strength of FIDLAR. It exploits the gradient-based planning and the differentiability of the trained `Evaluator` model for better optimization. During training, the gradient-based back-propagation from the `Evaluator` helps to reinforce the `Manager`.

All the DL-based versions of FIDLAR are several orders of magnitude faster than the (physics-based or GA-based) competitors while achieving improvement over other methods in flood mitigation. These characteristics allow us to entertain the possibility of real-time flood management, which was challenging for previous approaches.

Acknowledgments

This work is part of the Institute for Geospatial Understanding through an Integrative Discovery Environment (I-GUIDE) project, which is funded by the National Science Foundation under award number 2118329. Any opinions, findings, and conclusions or recommendations expressed herein are those of the authors and do not necessarily represent the views of funding sources.

References

Bai, S.; Kolter, J. Z.; and Koltun, V. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*.

Bentivoglio, R.; Isufi, E.; Jonkman, S. N.; and Taormina, R. 2022. Deep learning methods for flood mapping: a review of existing applications and future research directions. *Hydrology and Earth System Sciences Discussions*, 2022: 1–50.

Bharadhwaj, H.; Xie, K.; and Shkurti, F. 2020. Model-predictive control via cross-entropy and gradient-based optimization. In *Learning for Dynamics and Control*, 277–286. PMLR.

Bowes, B. D.; Tavakoli, A.; Wang, C.; Heydarian, A.; Behl, M.; Beling, P. A.; and Goodall, J. L. 2021. Flood mitigation in coastal urban catchments using real-time stormwater infrastructure control and reinforcement learning. *Journal of Hydroinformatics*, 23(3): 529–547.

Brody, S. D.; Zahran, S.; Maghelal, P.; Grover, H.; and Highfield, W. E. 2007. The rising costs of floods: Examining the impact of planning and development decisions on property damage in Florida. *Journal of the American Planning Association*, 73(3): 330–345.

Castro-Gama, M. E.; Popescu, I.; Li, S.; Mynett, A.; and van Dam, A. 2014. Flood inference simulation using surrogate modelling for the Yellow River multiple reservoir system. *Environmental modelling & software*, 55: 250–265.

Choubin, B.; Moradi, E.; Golshan, M.; Adamowski, J.; Sajedi-Hosseini, F.; and Mosavi, A. 2019. An ensemble prediction of flood susceptibility using multivariate discriminant analysis, classification and regression trees, and support vector machines. *Science of the Total Environment*, 651: 2087–2096.

Delaney, C. J.; Hartman, R. K.; Mendoza, J.; Dettinger, M.; Delle Monache, L.; Jasperse, J.; Ralph, F. M.; Talbot, C.; Brown, J.; Reynolds, D.; et al. 2020. Forecast informed reservoir operations using ensemble streamflow predictions for a multipurpose reservoir in Northern California. *Water Resources Research*, 56(9): e2019WR026604.

District, S. F. W. M. 2023. DBHYDRO of South Florida Water Management District. https://www.sfwmd.gov/science-data/dbhydro.

Fraehr, N.; Wang, Q. J.; Wu, W.; and Nathan, R. 2023. Development of a fast and accurate hybrid model for floodplain inundation simulations. *Water Resources Research*, 59(6): e2022WR033836.

Gidea, M.; and Katz, Y. 2018. Topological data analysis of financial time series: Landscapes of crashes. *Physica A: Statistical Mechanics and its Applications*, 491: 820–834.

Gomes Jr, M. N.; Rápalo, L. M.; Oliveira, P. T.; Giacomoni, M. H.; do Lago, C. A.; and Mendiondo, E. M. 2023. Modeling unsteady and steady 1D hydrodynamics under different hydraulic conceptualizations: Model/Software development and case studies. *Environmental Modelling & Software*, 167: 105733.

Jafarzadegan, K.; Moradkhani, H.; Pappenberger, F.; Moftakhari, H.; Bates, P.; Abbaszadeh, P.; Marsooli, R.; Ferreira, C.; Cloke, H. L.; Ogden, F.; et al. 2023. Recent advances and new frontiers in riverine and coastal flood modeling. *Reviews of Geophysics*, 61(2): e2022RG000788.

Jonkman, S. N.; and Vrijling, J. K. 2008. Loss of life due to floods. *Journal of Flood Risk Management*, 1(1): 43–56.

Jyothir, S.; Jalagam, S.; LeCun, Y.; and Sobal, V. 2023. Gradient-based Planning with World Models. *arXiv preprint arXiv:2312.17227*.

Kabir, M. H.; and Hossen, M. N. 2019. Impacts of flood and its possible solution in Bangladesh. *Disaster Adv*, 12(10): 48–57.

Karimanzira, D. 2016. Model based decision support systems. *Modeling, Control and Optimization of Water Systems: Systems Engineering Methods for Control and Decision Making Tasks*, 185–220.

Kerkez, B.; Gruden, C.; Lewis, M.; Montestruque, L.; Quigley, M.; Wong, B.; Bedig, A.; Kertesz, R.; Braun, T.; Cadwalader, O.; et al. 2016. Smarter stormwater systems.

Kipf, T. N.; and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

LeCun, Y. 2022. A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27. *Open Review*, 62(1).

Leon, A. S.; Kanashiro, E. A.; Valverde, R.; and Sridhar, V. 2014. Dynamic framework for intelligent control of river flooding: Case study. *Journal of Water Resources Planning and Management*, 140(2): 258–268.

Leon, A. S.; Tang, Y.; Qin, L.; and Chen, D. 2020. A MATLAB framework for forecasting optimal flow releases in a multi-storage system for flood control. *Environmental Modelling & Software*, 125: 104618.

Li, L.; Noorian, F.; Moss, D. J.; and Leong, P. H. 2014. Rolling window time series prediction using MapReduce. In *Proceedings of the 2014 IEEE 15th international conference on information reuse and integration (IEEE IRI 2014)*, 757–764. IEEE.

Medsker, L. R.; and Jain, L. 2001. Recurrent neural networks. *Design and Applications*, 5(64-67): 2.

Munawar, H. S.; Hammad, A.; Ullah, F.; and Ali, T. H. 2019. After the flood: A novel application of image processing and machine learning for post-flood disaster management. In *Proceedings of the 2nd International Conference on Sustainable Development in Civil Engineering (ICSDC 2019), Jamshoro, Pakistan*, 5–7.

O'Shea, K.; and Nash, R. 2015. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*.

Peker, İ. B.; Gülbaz, S.; Demir, V.; Orhan, O.; and Beden, N. 2024. Integration of HEC-RAS and HEC-HMS with GIS in flood modeling and flood hazard mapping. *Sustainability*, 16(3): 1226.

Rahman, M.; and Ali, M. S. 2024. Drivers of tidal flow variability in the Pussur fluvial estuary: A numerical study by HEC-RAS. *Heliyon*, 10(4).

Rivett, M. O.; Tremblay-Levesque, L.-C.; Carter, R.; Thetard, R. C.; Tengatenga, M.; Phoya, A.; Mbalame, E.; Mchilikizo, E.; Kumwenda, S.; Mleta, P.; et al. 2022. Acute health risks to community hand-pumped groundwater supplies following Cyclone Idai flooding. *Science of The Total Environment*, 806: 150598.

Ruder, S. 2016. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*.

Sadler, J. M.; Goodall, J. L.; Behl, M.; Bowes, B. D.; and Morsy, M. M. 2020. Exploring real-time control of stormwater systems for mitigating flood risk due to sea level rise. *Journal of Hydrology*, 583: 124571.

Sadler, J. M.; Goodall, J. L.; Behl, M.; Morsy, M. M.; Culver, T. B.; and Bowes, B. D. 2019. Leveraging open source software and parallel computing for model predictive control of urban drainage systems using EPA-SWMM5. *Environmental Modelling & Software*, 120: 104484.

Saha, A.; Pal, S. C.; Arabameri, A.; Blaschke, T.; Panahi, S.; Chowdhuri, I.; Chakrabortty, R.; Costache, R.; and Arora, A. 2021. Flood susceptibility assessment using novel ensemble of hyperpipes and support vector regression algorithms. *Water*, 13(2): 241.

Schwanenberg, D.; Becker, B.; and Xu, M. 2015. The open real-time control (RTC)-Tools software framework for modeling RTC in water resources sytems. *Journal of Hydroinformatics*, 17(1): 130–148.

Shi, J.; Myana, R.; Stebliankin, V.; Shirali, A.; and Narasimhan, G. 2023a. Explainable parallel rcnn with novel feature representation for time series forecasting. In *International Workshop on Advanced Analytics and Learning on Temporal Data*, 56–75. Springer.

Shi, J.; Stebliankin, V.; Wang, Z.; Wang, S.; and Narasimhan, G. 2023b. Graph Transformer Network for Flood Forecasting with Heterogeneous Covariates. *arXiv preprint arXiv:2310.07631*.

Shi, J.; Yin, Z.; Myana, R.; Ishtiaq, K.; John, A.; Obeysekera, J.; Leon, A.; and Narasimhan, G. 2023c. Deep Learning Models for Water Stage Predictions in South Florida. *arXiv preprint arXiv:2306.15907*.

Suykens, J. A.; Vandewalle, J. P.; and De Moor, B. L. 1995. *Artificial neural networks for modelling and control of non-linear systems*. Springer Science & Business Media.

Tanim, A. H.; McRae, C. B.; Tavakol-Davani, H.; and Goharian, E. 2022. Flood detection in urban areas using satellite imagery and machine learning. *Water*, 14(7): 1140.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wing, O. E.; Lehman, W.; Bates, P. D.; Sampson, C. C.; Quinn, N.; Smith, A. M.; Neal, J. C.; Porter, J. R.; and Kousky, C. 2022. Inequitable patterns of US flood risk in the Anthropocene. *Nature Climate Change*, 12(2): 156–162.

Wu, X.; Guo, J.; Wu, X.; and Guo, J. 2021. A new economic loss assessment system for urban severe rainfall and flooding disasters based on big data fusion. *Economic impacts and emergency management of disasters in China*, 259–287.

Yan, J.; Jin, J.; Chen, F.; Yu, G.; Yin, H.; and Wang, W. 2018. Urban flash flood forecast using support vector machine and numerical simulation. *Journal of Hydroinformatics*, 20(1): 221–231.

Yang, S.-N.; and Chang, L.-C. 2020. Regional inundation forecasting using machine learning techniques with the internet of things. *Water*, 12(6): 1578.

Yin, J.; Gao, Y.; Chen, R.; Yu, D.; Wilby, R.; Wright, N.; Ge, Y.; Bricker, J.; Gong, H.; and Guan, M. 2023a. Flash floods: why are more of them devastating the world's driest regions? *Nature*, 615(7951): 212–215.

Yin, Z.; Bian, L.; Hu, B.; Shi, J.; and Leon, A. S. 2023b. Physic-informed neural network approach coupled with boundary conditions for solving 1D steady shallow water equations for riverine system. In *World Environmental and Water Resources Congress 2023*, 280–288.

Zarei, M.; Bozorg-Haddad, O.; Baghban, S.; Delpasand, M.; Goharian, E.; and Loáiciga, H. A. 2021. Machine-learning algorithms for forecast-informed reservoir operation (FIRO) to reduce flood damages. *Scientific reports*, 11(1): 24295.

Zhang, Z.; and Dong, Y. 2020. Temperature forecasting via convolutional recurrent neural networks based on time-series data. *Complexity*, 2020: 1–8.

Zhou, Y.; Wu, W.; Nathan, R.; and Wang, Q. J. 2021. A rapid flood inundation modelling framework using deep learning with spatial reduction and reconstruction. *Environmental Modelling & Software*, 143: 105112.