Deep Learning Methodology for Scalable Emulation of Hydrodynamic Flood Simulations Across Diverse Terrains and Conditions

Lana Khalifa, Barak Fishbain

Abstract

This study presents a deep learning methodology for emulating hydrodynamic flood simulations, utilizing a standard UNet architecture to achieve high accuracy and robust generalization across diverse, complex datasets. A novel closure model is introduced, aggregating patch-level predictions to reconstruct full domain-level flood forecasts. The approach demonstrates substantial scalability, outperforming baseline models, especially as flood events evolve over time. While challenges remain in predicting early-stage flooding, the method reliably adapts to various domain sizes and conditions. These results emphasize the potential of deep learning—particularly the UNet architecture—as an efficient, scalable, and practical tool for operational flood forecasting. Future improvements, such as enhancing early-stage predictions and integrating further physical constraints, are anticipated to boost the model's performance even further, reinforcing its suitability for real-time, high-resolution flood prediction and management across a wide range of terrains and hydrologic scenarios.

1 Introduction

1.1 Problem Statement

Between 2000 and 2019, floods were the most frequent natural disaster and the fourth leading cause of death among natural disasters (CRED). To mitigate their impact, it’s essential to forecast floods by simulating water depths and velocities, providing timely warnings and improving preparedness.

In pursuit of improved flood prediction, both empirical data-driven approaches and physics-based models have been explored, each relying on historical datasets. Broadly, empirical methods offer substantial computational efficiency but are constrained by limited predictive accuracy. In contrast, physically based two-dimensional (2D) and three-dimensional (3D) hydrodynamic models, which

solve governing physical equations, deliver enhanced accuracy but at the cost of significant computational demands (Teng et al., 2017). This restricts their practical application to smaller regions, typically to the scale of 1,000 km² with raster grid resolutions typically coarser than 10 m for 2D simulations. Therefore, advancing methods that reduce the computational burden of physical models is essential to enable high-resolution, large-scale, and real-time flood simulations. Machine learning (ML) offers a promising alternative to traditional models by significantly improving computational efficiency (Mosavi et al., 2018). As surrogate models, ML can emulate physically based simulations if provided with enough data and an appropriate learning strategy. ML learns mappings from inputs (e.g., terrain, hydrologic events) to outputs (e.g., water depth) without explicit physical laws, training by minimizing a loss function and validating on unseen data. Neural Networks (NNs), a type of ML inspired by the brain, include Deep Learning (DL) models with multiple layers to capture complex patterns. Multi-Layer Perceptrons (MLPs) have fully connected neurons, while Convolutional Neural Networks (CNNs) process spatial data using convolution kernels. Still, NNs are only as good as the data they are trained on, facing challenges like data scarcity and limited generalization and computational complexity. Here we present a DL-based methodology for hydrological modeling that is computationally efficient and relies on significantly less data through augmentation, rendering this method as scalable.

1.2 Prior Art

A comprehensive review of the literature reveals several notable contributions to the application of machine learning in flood modeling (Bermúdez et al., 2019; Chang et al., 2010; Elkhrachy, 2022; Guo et al., 2021, 2022; Hosseiny et al., 2020; Kabir et al., 2020; Löwe et al., 2021; Pan et al., 2011). To this end, Elkhrachy (2022) conducted simulations on a single terrain and flood event, employing remotely sensed data to predict maximum water depth in New Cairo. This scope has been expanded by encompassing multiple flooding events on a single terrain, frequently utilizing Multi-Layer Perceptron (MLP) models that do not explicitly account for spatial relationships (Bermúdez et al., 2019; Chang et al., 2010; Hosseiny et al., 2020; Kabir et al., 2020; Pan et al., 2011). Guo et al. (2021, 2022) introduced a convolutional neural network (CNN) encoder-decoder framework trained on spatial patches from various terrains. The earlier study (Guo et al., 2021) investigated generalization to differing rainfall intensities without evaluating performance on new

patches, whereas the later work (Guo et al., 2022) demonstrated terrain generalization using the same flood event across samples. Löwe et al. (2021) further extended generalization to encompass diverse terrains and rainfall scenarios. However, their U-FLOOD model treated spatial patches independently, neglected inter-patch flow dynamics, assumed invariably dry initial conditions, and was restricted to predicting only maximum water depths.

Of the above studies several incorporated temporal sequences of precipitation data as model inputs (Guo et al., 2021, 2022; Kabir et al., 2020; Pan et al., 2011), whereas others (Bermúdez et al., 2019; Hosseiny et al., 2020; Löwe et al., 2021) utilized summary statistics. The majority of the reviewed models focused on predicting maximum water depths, with only a select few (Chang et al., 2010; Kabir et al., 2020; Pan et al., 2011) extending their predictive scope to include dynamic flood states across temporal dimensions.

A review of broader machine learning applications, such as those in flow modeling and heat conduction, reveals parallel limitations. Certain methodologies adopt pointwise prediction strategies based on spatial coordinates (Ang et al., 2023; Cheng & Zhang, 2021), while others employ raster-based inputs yet lack dynamic boundary conditions, internal condition variability, or the integration of auxiliary data. For example, Alhada-Lahbabi et al. (2023) implemented a UNet architecture to predict polarization dynamics solely on the basis of initial conditions. Similarly, Chen et al. (2023) trained convolutional neural networks utilizing auxiliary material data. Notably, Obiols-Sales et al. (2020) distinctively appended boundary conditions to input fields, whereas Nguyen et al. (2023) advanced the integration of initial conditions, boundary conditions, and auxiliary data, although their approach was limited to a single material and static boundary conditions.

Collectively, these studies underscore both the promise and inherent limitations of deep learning approaches in hydrological modeling. Notably, several analyses are constrained by data limited to a single terrain (Alhada-Lahbabi et al., 2023), highlighting the need for more robust methodologies capable of generalizing across a broader range of hydrologic and geomorphologic conditions. Even when considering the broader scope of ML in flow modeling it does show promise, but key gaps remain. Many models lack generalization to dynamic boundary conditions, and diverse terrains, or fail to preserve temporal dynamics. A comprehensive ML model for flood prediction should integrate dynamic boundary conditions, spatially interconnected domains, and generalize across terrains, hydrologic events, and time.

2 Methodology

2.1 Overview of the Approach

This study delineates a comprehensive methodological pipeline for the emulation of hydrodynamic flood simulations utilizing deep learning frameworks. The approach is structured around four principal components: (1) the generation of training data via HEC-RAS 2D flood simulations conducted across a suite of arid terrain types; (2) the transformation of simulation outputs into temporally resolved, patch-based datasets for water depth prediction; (3) the systematic training and evaluation of multiple deep learning architectures, each tasked with forecasting future water depths *at the patch scale*; and (4) the deployment of an iterative closure procedure to aggregate patch-level predictions into domain-scale flood maps. Collectively, this methodology facilitates the rapid and accurate modeling of flood events.

It is important to note that stage (3) aims at finding, for any given time, t, the water level at any given patch in a future time, $t + \Delta t$. To achieve this, the model is trained based on the current water level in each patch and the *future* water level and boundary conditions. Once trained, given a patch's current water level and future boundary conditions, the model can obtain the water level, in a future time, within the patch. However, the future boundary conditions are unknown at the time of prediction. This is addressed by the closure model, which uses the very same trained AI architecture and is elaborated on in Section 2.4.

2.2 Simulations Setup

The two-dimensional unsteady flow module of HEC-RAS (Hydrologic Engineering Center, 2021) was utilized to perform flash flood simulations. Each simulation incorporated 1-meter resolution Digital Elevation Models (DEMs) corresponding to unique terrain selections with varying spatial extents (see Figure 1a). The computational domain for each simulation was discretized into a regular grid comprising 10×10-meter cells, within which HEC-RAS computed water surface elevations using a sub-grid bathymetric approach (Figure 1b). This entailed referencing an underlying matrix of 11 × 11 cells at 1-meter resolution to accurately represent topographic variability (Figure 1c).

For model initialization, all terrain domains were assumed to be dry, and precipitation was intentionally excluded to isolate the effects of hydrodynamic forcing. Hydrographs were assigned randomly to predefined inlet locations, while normal depth conditions served as downstream boundary specifications unless otherwise indicated. Each simulation was conducted over a temporal window of 4.5 hours, with water depth outputs recorded at one-minute intervals. The Diffusion Wave Approximation was employed throughout, prioritizing numerical efficiency and independence from explicit velocity calculations. This approach was selected to enable the deep learning framework to focus predominantly on the emulation of water surface elevation dynamics.

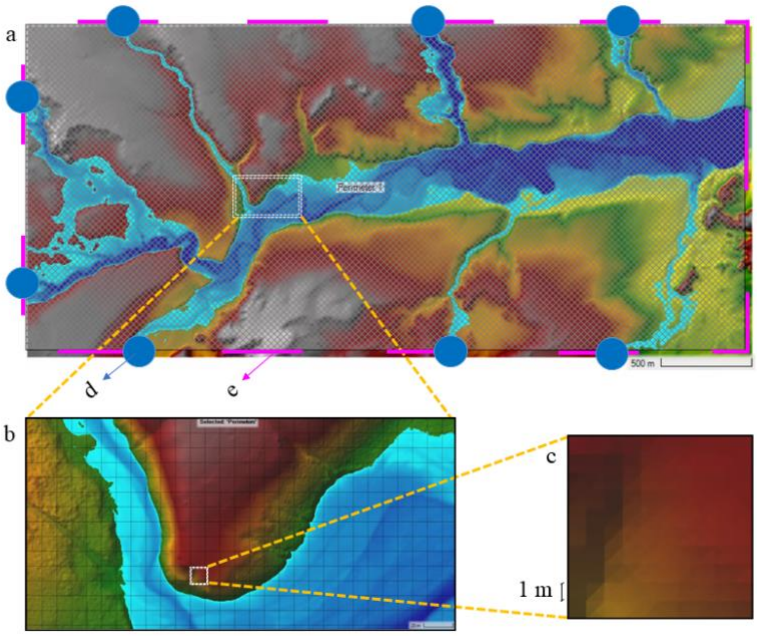

Figure 1 - Setup for a Single Simulation, including Terrain, Grid, Sub-Grid Terrain Representation, and Water Inlet Placement. Panel a shows the overall terrain with the 1-meter Digital Elevation Model (DEM), representing the entire simulation area. Panel b illustrates the structured computational grid. Panel c provides a close-up of one computational cell (10 × 10 m), displaying the underlying 11 × 11 array of 1-meter resolution elevation points used for sub-grid bathymetry, which HEC-RAS utilizes to calculate water surface elevations. d and e arrows show the placement of water inlets—manually and automatically (cyclically), respectively.

2.2.1 Terrains

High-resolution Digital Elevation Models (DEMs) at a 1-meter spatial resolution were sourced from the United States Geological Survey (USGS) 3D Elevation Program (2025). To ensure the selection of exposed riverbeds representative of arid environments, DEMs acquired during the summer months (June–August) were intersected with dry climate ecoregions as delineated by the United States Environmental Protection Agency (2015), encompassing the North American Deserts, Southern Semi-Arid Highlands, and the Mediterranean California region. From these

areas of intersection, four study sites (designated prj_01 through prj_04) were identified for subsequent analysis.

The selection methodology is visualized in Figure 2, which depicts the spatial overlap between dry climate ecoregions (indicated by hatched areas), high-quality summer-acquired DEMs (highlighted as yellow polygons), and intermittent river networks (depicted as blue lines). The four study sites, outlined in red, were chosen based on their location within these intersecting regions. The right panel of Figure 2 presents a detailed view of study area prj_03, illustrating the distribution of selected terrain tiles (represented by grayscale squares) with respect to the mapped intermittent river channels.

Within each designated study area, rectangular terrain tiles containing intermittent rivers were delineated and subsequently verified using the Food and Agriculture Organization (FAO) Catalog of Rivers of North America (United Nations, 2025). The selected terrain tiles were then clipped from the broader DEMs for use in hydrodynamic simulations. The right panel of Figure 2 exemplifies this procedure for prj_03, providing a representative visualization of the dataset construction process.

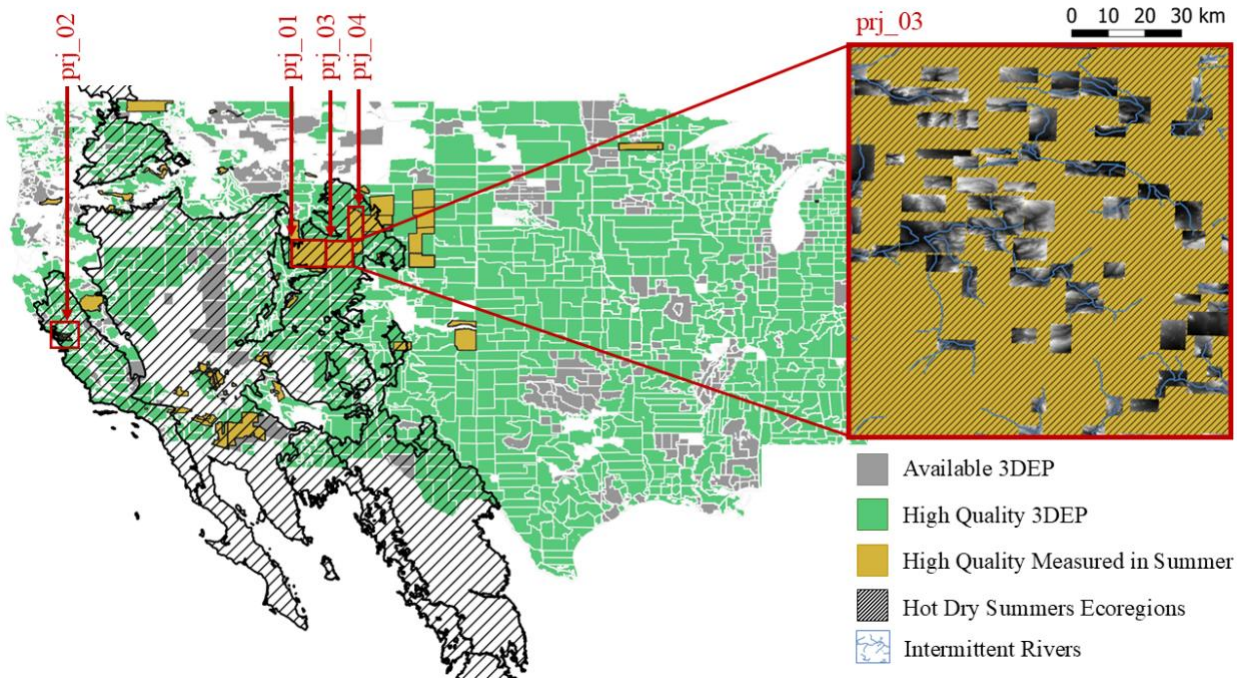

Figure 2 - Selection of Study Areas Based on the Overlap of Dry-Climate Ecoregions, Summer-Acquired Elevation Data, and Intermittent Rivers. The map shows the spatial intersection between dry climate ecoregions (hatched areas), high-quality 1-meter DEMs acquired in summer (yellow polygons), and intermittent rivers (blue lines). Four study areas (prj_01 to prj_04) were

identified within regions of overlap and are outlined in red. The right panel provides a zoomed-in view of prj_03, illustrating the terrain tiles selected (grayscale squares) in relation to intermittent river channels.

2.2.2 Hydrographs

Nine hydrographs with low initial discharges—selected to avoid numerical artifacts in HEC-RAS were obtained from NOAA's FLASH database (2025) and is depicted in Figure S1 in the Supplementary Material. The 4.5-hour simulation window captured only the rising limb of the hydrographs.

2.2.3 Execution

Simulation configurations varied across projects. In prj_01, a total of 42 simulations were conducted, wherein hydrographs were manually positioned at the upstream boundary (as indicated by blue circles in Figure 1a), with a normal depth condition applied at the downstream boundary. Conversely, for prj_02 (38 simulations), prj_03 (85 simulations), and prj_04 (45 simulations), outlet conditions were omitted entirely. In these latter projects, hydrographs were either manually placed upstream (prj_02) or automatically distributed along the domain perimeter in a cyclic manner (prj_03 and prj_04; see pink lines in Figure 1a). While prj_01 simulations utilized nine distinct hydrographs, a single hydrograph was repetitively applied across all inlets in prj_02–04.
To minimize boundary artifacts—such as downstream water accumulation in the absence of an outlet or upstream flow when a hydrograph was applied at the downstream boundary—all simulation outputs were trimmed by 1 km inward from the domain edges. Temporal sampling was performed at four predefined pairs of time points, each separated by a 60-minute prediction horizon: $(t_1, t_1+60) = (0, 60)$ minutes; $(t_2, t_2+60) = (70, 130)$ minutes; $(t_3, t_3+60) = (150, 210)$ minutes; and $(t_4, t_4+60) = (210, 270)$ minutes. Corresponding water depth maps for these time points were post-processed to generate spatial data patches for use in model training. Patches derived from t_1 were directly incorporated into the training database. For t_2, t_3, and t_4, augmentation techniques—namely, rotation and mirroring—were employed to increase sample diversity. This process is illustrated in Figure 3). In addition, patches at t_2 and t_4 were translated by 16 pixels in both spatial dimensions relative to those at t_1 and t_3. Identical augmentation strategies were applied to the associated future time steps ($t_1 + \Delta t$, $t_2 + \Delta t$, $t_3 + \Delta t$, and $t_4 + \Delta t$) and terrain maps.

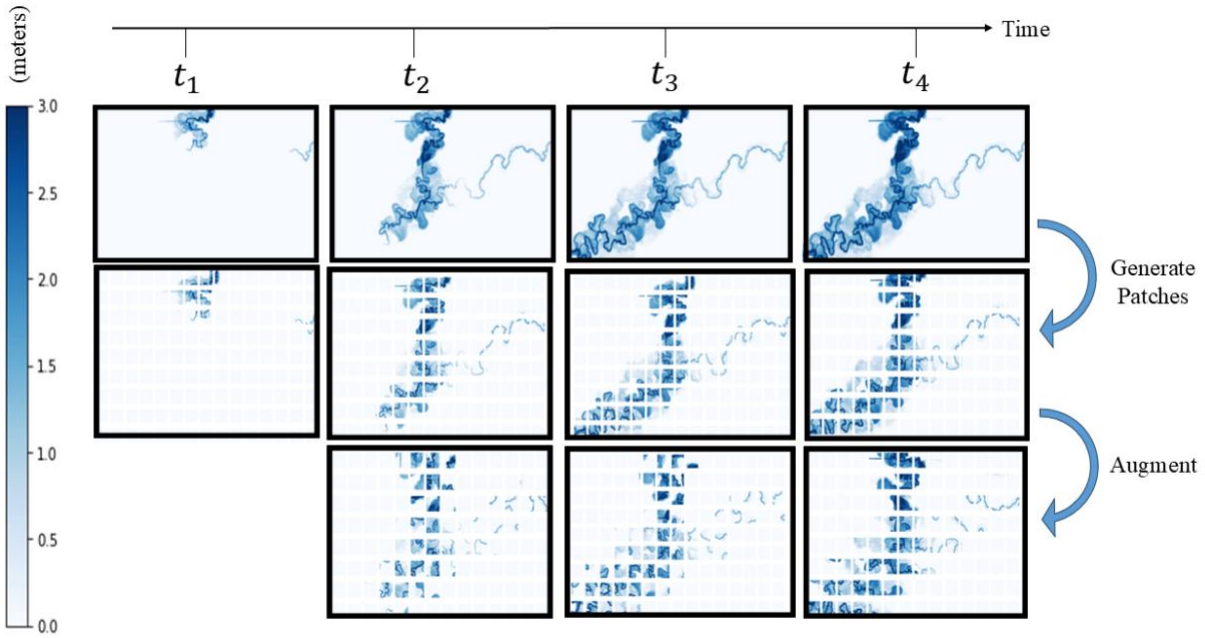

Figure 3 – Terrain augmentation techniques—rotation and mirroring were employed to increase sample diversity. In addition, patches at t_2 and t_4 were translated by 16 pixels in both spatial dimensions relative to those at t_1 and t_3.

This methodology was systematically applied to all 210 simulations. Each resulting sample comprised a terrain input (321 × 321 pixels), water depth at time t (32 × 32 pixels), and boundary water depth at time t + Δt (32 × 32 pixels), where only boundary pixels exhibited non-zero values. These were provided as input to the neural network. The target for prediction was defined as the difference in water depth between t + Δt and t (see Figure 4). Persistently dry patches were excluded from the dataset, and terrain elevations within each patch were shifted so the minimum height in the patch was set to zero. The entire computational pipeline—including conversion of HEC-RAS HDF output files and terrain TIFF inputs to neural network training samples—is thoroughly documented and accessible on GitHub[1].

[1] https://github.com/LanaKhalifa/deep-learning-flood-modeling/tree/main/simulations_to_samples

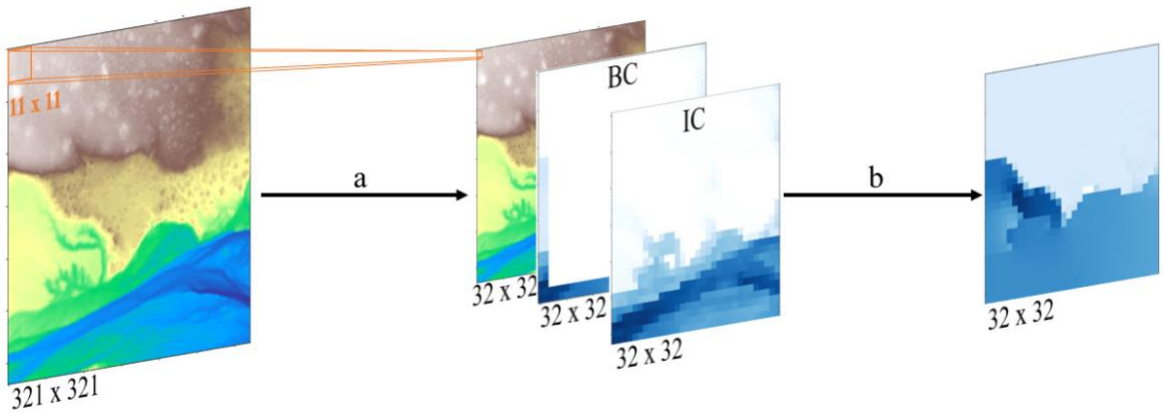

Figure 4 - Terrain down sampling pipeline. Translation (a) The 321×321 terrain is down sampled to 32×32 using the down sampling module that mimics HEC-RAS's sub-grid logic, where each output pixel represents an underlying 11×11 region. The resulting 32×32 terrain is then concatenated with the initial condition (IC) and boundary condition (BC) maps along the channel dimension. Translation (b) - the combined input is passed to the main architecture, to predict the water depth difference

2.3 Deep Learning Architectures

A variety of deep learning architectures were implemented to predict changes in water depth for each patch. The architectures examined are henceforth referred to by their abbreviated designations. The 'Simplified UNet' architecture presented by Alhada-Lahbabi et al. (2023) was chosen to represent UNet-based approaches, while the 'Classic UNet' from Ronneberger et al. (2015) served as a performance baseline. An 'Encoder-Decoder with Self-Attention' model, as described by Chen et al. (2023), was also implemented, alongside a variant termed 'Non-down-sampling Convolutions with Self-Attention.' This variant omits the encoder-decoder structure, instead employing six convolutional layers with 3×3 kernels and unit stride, and incorporates a self-attention mechanism at the network's midpoint. This design maintains full spatial resolution throughout the architecture, aligning closely with the cell-wise computations characteristic of HEC-RAS. Furthermore, a 'Non-down-sampling Convolutions' architecture, which excludes self-attention, was evaluated to isolate the specific contributions of convolutional operations. A 'Modified UNet with ResNet,' based on the work of Santos et al. (2020), was included to assess the impact of residual blocks. An 'Encoder-

Decoder with Large Convolutions' from Obiols-Sales et al. (2020) completed the set of evaluated models. All architectures are available within the models repository on GitHub[2].

Since the base models lacked terrain down-sampling mechanisms for auxiliary data, a dedicated down-sampler was introduced to reduce the 321×321 terrain data to 32×32 prior to concatenation with the initial condition (IC) and boundary condition (BC) maps along the channel dimension (refer to Figure 4). Two down-sampling methods were developed to emulate HEC-RAS's sub-grid logic, ensuring that each output pixel appropriately represents the underlying 11×11 region from the original terrain grid. The first method, termed "alter," utilized three convolutional layers (a 3×3 kernel with stride 2, a custom alternating-stride layer switching between strides of 2 and 3, and a final 2×2 layer with stride 2). The second method (dubbed "k11s10p0") employed a single 11×11 kernel with a stride of 10 and no padding. For comparative purposes, additional models were trained using bicubic interpolation and, separately, without terrain input.

Three primary dataset configurations were constructed. The small_train and small_val sets comprised seven simulations from prj_01, specifically curated for rapid experimentation and model comparison. The big_train and big_val datasets incorporated all simulations from all projects, except for seven simulations per project, which were reserved for the big_test dataset for testing. Analogous versions, prj_01_train_val and prj_01_test, were constructed using only prj_01 data and served as a benchmark for model performance given the high-quality and hand-curated nature of these simulations.

The training approach was organized into three methodical stages to systematically identify the optimal architectural configurations. In Stage 1, the custom 'Non-down-sampling Convolutions with Self-Attention' architecture was fine-tuned on small_train, with systematic exploration of dataset parameters, architectural variants, and optimization strategies; improvements in small_val loss were incrementally retained via a greedy optimization protocol. Stage 2 involved training all candidate architectures on small_train for 300 epochs utilizing the L1 loss function (as specified in Equation 1), employing the optimized settings from Stage 1, and subsequently selecting the top four performers based on small_val outcomes. In Stage 3, the top four architectures from Stage 2 underwent retraining on big_train and validation on big_val to determine the final model. The

[2] https://github.com/LanaKhalifa/deep-learning-flood-modeling/tree/main/multi_architecture_training/models

complete implementation of this multi-stage process is accessible in the multi_architecture_training module on GitHub[3].

$$Loss = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{M}\sum_{j=1}^{M}\left|\, y_{i,j} - \hat{y}_{i,j}\right|\right) \qquad \textit{Equation 1}$$

The architecture demonstrating the highest performance in Stage 3 was further evaluated using the Relative Absolute Error (RAE) metric. For N the number of pixels in each sample (here $N = 1024 = 32^2$ pixels); $\hat{y}_i$, the predicted value at pixel i by the trained model; $\hat{y}_i^{dummy}$, the predicted value at pixel i by a dummy model; y_i: the ground truth value at pixel i, the RAE is given by:

$$RAE = \frac{\sum_{i=1}^{M}|y_i - \hat{y}_i\,|}{\sum_{i=1}^{M}|y_i - \hat{y}_i^{dummy}|} \qquad \textit{Equation 2}$$

The RAE metric quantifies the improvement of the model relative to a baseline (dummy) model that assumes no temporal variation in water depth (steady-state). RAE values were calculated for each sample across all major datasets: big_train, big_val, big_test, prj_01_train_val, and prj_01_test. For visualization purposes, instances where RAE was infinite were excluded from boxplot analysis. Detailed scripts for model evaluation and visualization are provided within the evaluate_and_visualize_best_model directory on GitHub.

2.4 Closure model

In the application of a trained deep learning architecture for domain-wide hydrodynamic prediction, the subsequent methodology entails propagating model forecasts across the entire spatial domain, utilizing the prescribed boundary conditions (BCs) as constraints. A significant methodological challenge lies in the specification of BCs for internal patches—termed internal BCs—which, by their nature, remain indeterminate prior to prediction.

To address this issue, the approach initializes internal BCs using the water depth values from the current temporal state (i.e., the initial conditions), thus providing a pragmatic estimate. The model subsequently performs a forward inference pass, updating water depth predictions throughout the

[3] https://github.com/LanaKhalifa/deep-learning-flood-modeling/tree/main/multi_architecture_training

domain. For all cells on internal boundaries, the updated predictions are then recursively supplied as boundary conditions for ensuing iterations.

In each iteration, patch positions within the spatial domain are systematically shifted such that internal BCs are centrally located within their respective patches. Additional model passes are conducted to iteratively refine the water depth predictions, with spatial patch locations being continuously adjusted. This iterative process proceeds until the mean change in predicted water depths between successive cycles falls below a pre-defined tolerance, indicating convergence. Here, four distinct patch traversal sequences (as depicted in Figure 5) are employed in rotation throughout the iterative process. The convergence criterion is rigorously defined by the mean absolute variation in predicted water depths between consecutive four-sequence cycles. Comprehensive implementation details of the closure model are available in the associated GitHub repository[4].

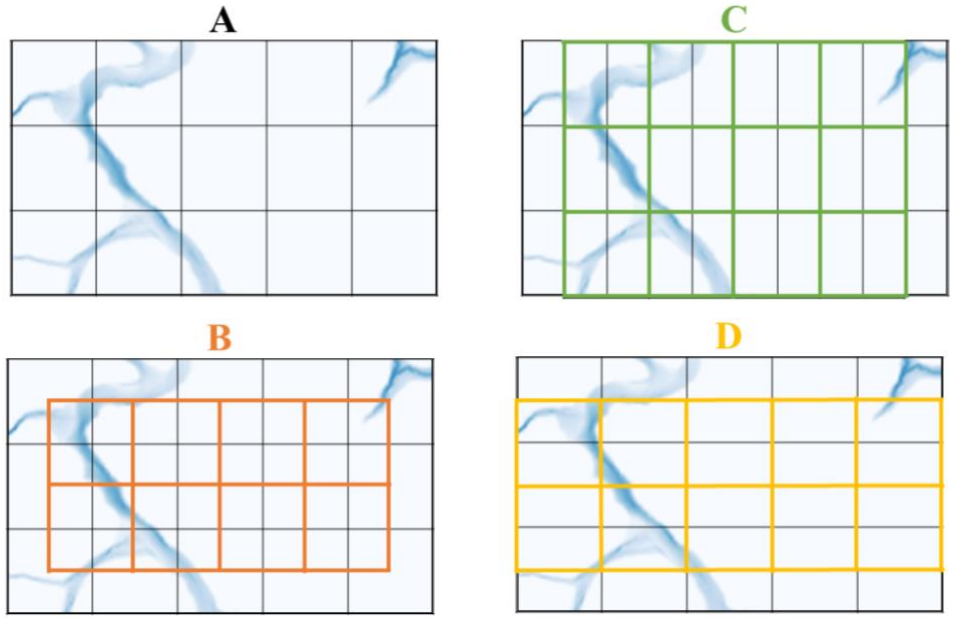

Figure 5 - Patch Ordering (Types A–D) Used During Iterative Domain-Wide Prediction for Closure

3 Results

At first a greedy optimization of the 'Non-down-sampling Convolutions with Self-Attention' architecture identified optimal configurations from 47 variants on the small_train/small_val datasets (Supporting

[4] https://github.com/LanaKhalifa/deep-learning-flood-modeling/tree/main/full_domain_closure_best_model

Information Table S1). Key parameters for the model included initial water depth and boundary conditions, with a 2-pixel boundary width, 60-minute prediction intervals, Float64 precision, zero-shifted terrain elevations, 'alter' downsampler, LeakyReLU activation, Kaiming initialization, and scheduled ADAM optimization. All architectures trained for 300 epochs using the above configurations. The train and validation loss function values with respect to the number of epochs is given in Figure 6.

Overall, the Classic UNet has produced the best results with a minimum loss (Equation 1) of 0.0176. The other methods provided comparable results where the 'Non-down-sampling Convolutions with Self-Attention' has resulted in a minimum loss of 0.0277; the 'Non-down-sampling Convolutions' produced a loss of 0.0495; and the 'Encoder-Decoder with Self-Attention' produced a loss of 0.0509. Remaining architectures performed substantially worse but exceeded the dummy baseline (0.173).

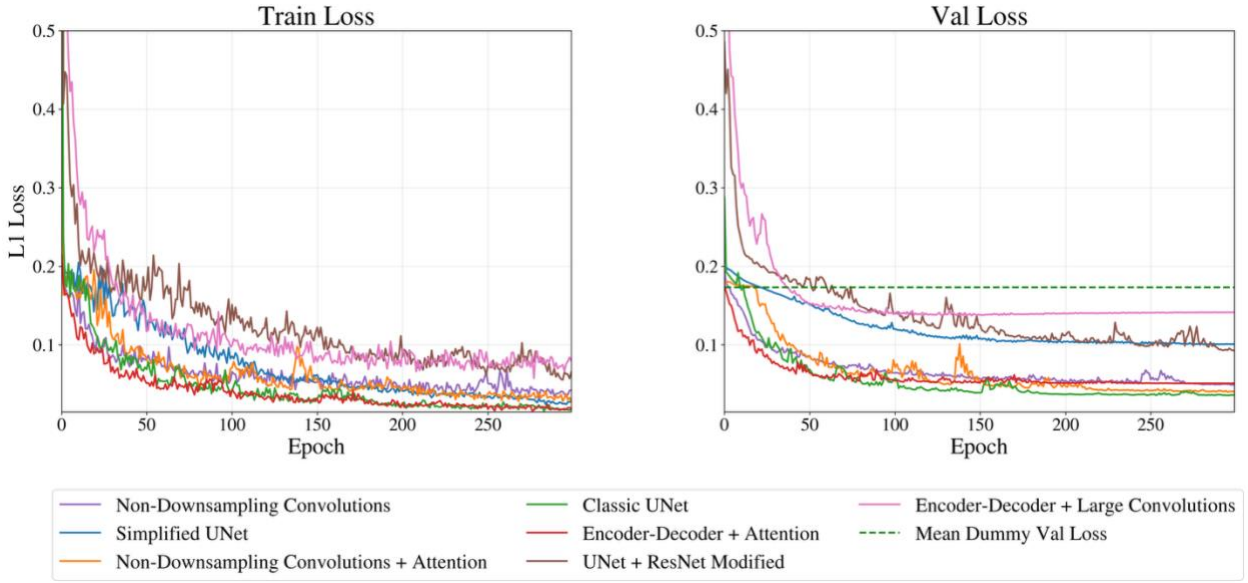

Figure 6 - Training and Validation Loss Function Values of all Architectures with Respect to the Epoch Number, Trained on the Small Dataset.

When trained on the big_set, the top four architectures again showed the Classic UNet as the best performer, followed closely by the 'Non-down-sampling Convolutions with Self-Attention' (Figure 7). Based on this analysis, the Classic UNet was selected as the production model for subsequent evaluation and closure model implementation.

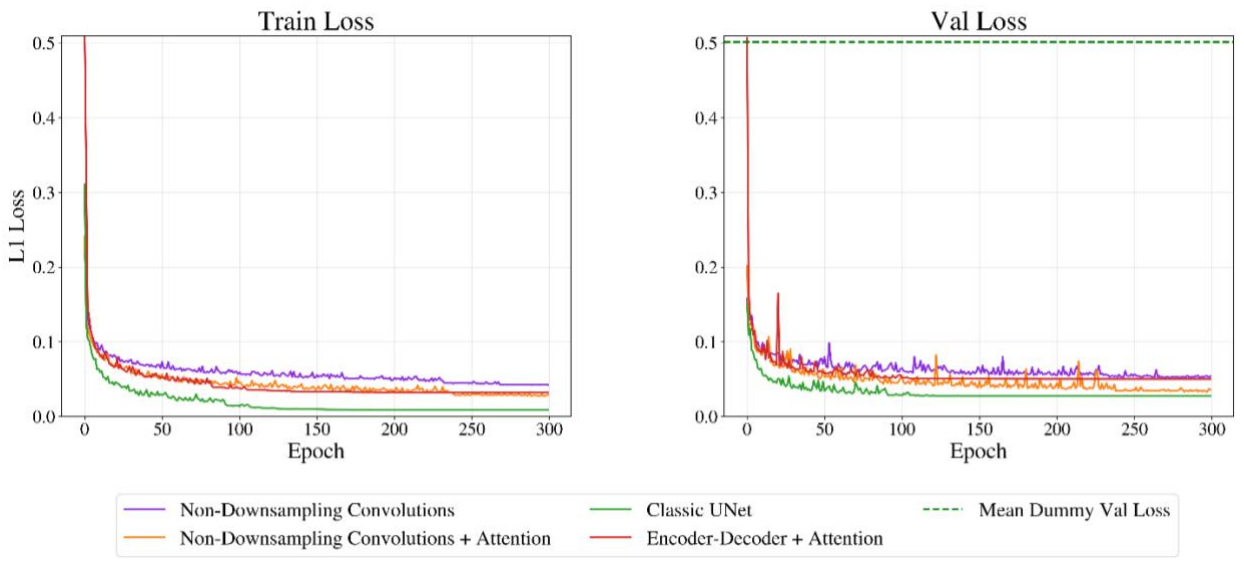

Figure 7 - Training and Validation Loss Function Values of Best Four Architectures with Respect to the Epoch Number, Trained on the Big Dataset

Figure 8 presents the Classic UNet RAE values across the big and the hand curated (prj_01) datasets. On big_test, the model achieved a median RAE of 0.11, representing a ten-fold improvement over the dummy baseline. Distinct median RAE values and varying interquartile ranges were observed across different datasets; however, all non-outlier RAE values remained below 0.6. The prj_01_test subset demonstrated comparable median RAE and IQR distributions to big_test.

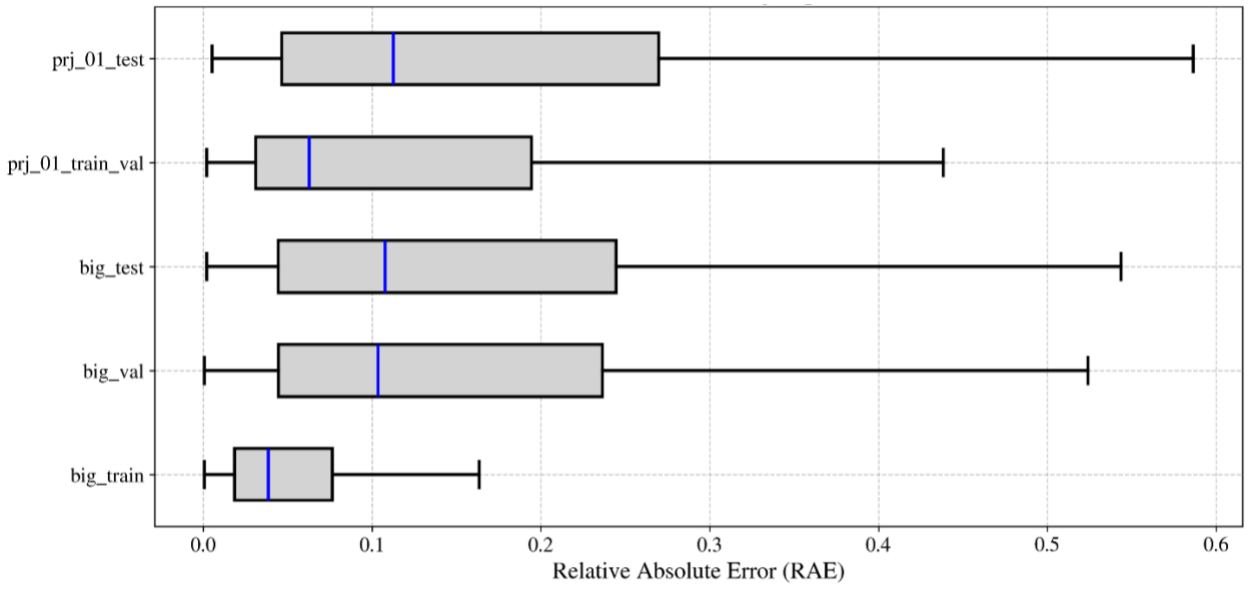

Figure 8 - Classic UNET RAE for big and prj_01 dataset

Figure 9 displays model predictions for chosen complex patches from the prj_01_test dataset. The first column (from the left) presents the model's predictions. Second column depicts the ground truth, where the

third column is the difference error. The right column shows the distribution of the error for the patch in question. To ensure an unbiased assessment, predictions for 270 randomly sampled instances from prj_01_test are presented in Supplementary Information Figures S2–S4. The model precisely reproduced dry static regions and captured fine-scale hydrodynamic variations in wet areas. Error distributions consistently showed near-zero mean values with symmetric distributions.

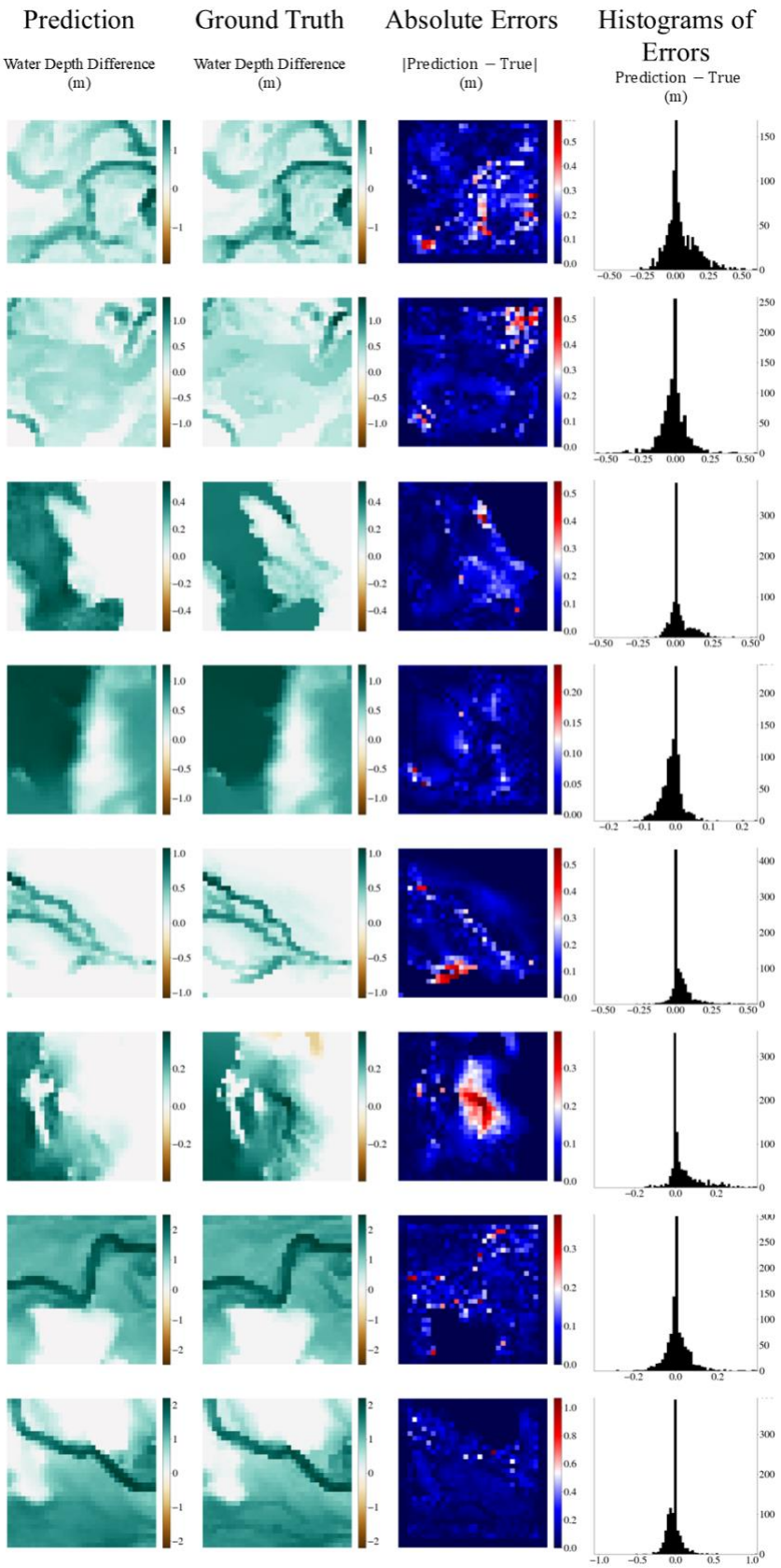
Prediction
Water Depth Difference (m)
Ground Truth
Water Depth Difference (m)
Absolute Errors
|Prediction − True| (m)
Histograms of Errors
Prediction − True (m)

Figure 9 – Example of Patch level results. First column (from the left) presents the model's predictions. Second column depicts the ground truth, where the third column is the difference. The rightest column shows the distribution of the error for the patch in question.

The closure model was tested on simulations from prj_01–prj_04, which were excluded from training. Figure 10 shows the RAE per time snapshot, color-coded by time (t_1 = 0 min, t_2 = 70 min, t_3 = 140 min, t_4 = 210 min) and scaled by domain size. The lowest median RAE was observed at t_4 (0.42), followed by t_3 (0.64), while t_1 (0.90) and t_2 (0.92) showed significantly higher errors—approaching the dummy baseline. No clear correlation was found between domain size and RAE.

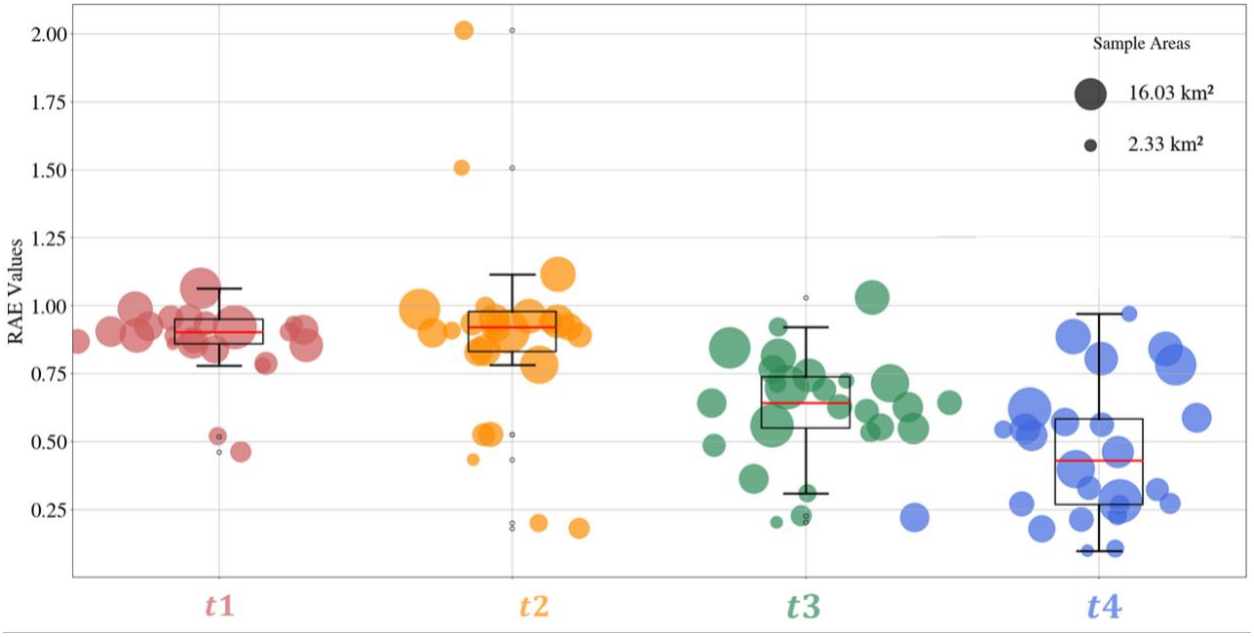

Figure 10 - RAE values per time snapshot (t_1 = 0 min, t_2 = 70 min, t_3 = 140 min, t_4 = 210 min) and domain size for prj_01–prj_04. The lowest median RAE occurred at t_4 (0.42), then t_3 (0.64), while t_1 (0.90) and t_2 (0.92) were higher and near the baseline. No clear link was found between domain size and RAE.

Figure 11 shows full-domain predictions for one simulation and snapshot, with an RAE of 0.27. Below the prediction, convergence behavior is evaluated using two metrics: the mean absolute difference between successive predictions (stopping criterion) and the mean absolute error relative to the ground truth. The latter is compared against a dummy baseline to contextualize model performance.

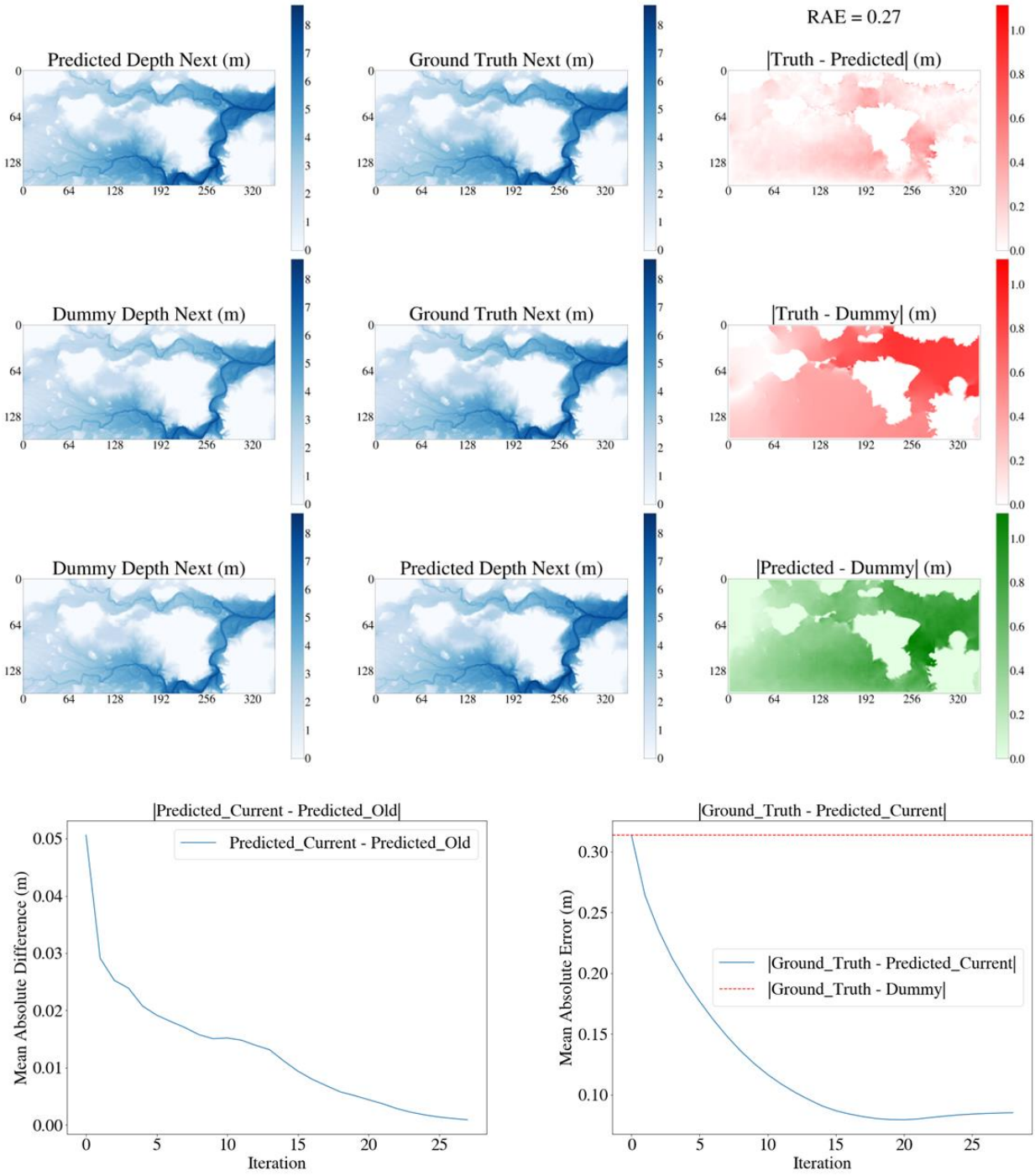

Figure 11 - Overall model's performance

4 Discussion

The present study introduces a deep learning framework tailored for the emulation of hydrodynamic flood simulations, specifically focusing on forecasting future water depth distributions across heterogeneous terrain configurations, boundary conditions, and temporal scenarios. Employing the Classic UNet architecture, the approach demonstrated notable superiority at the patch level, consistently outperforming alternative models incorporating self-attention mechanisms and variants lacking down sampling operations. The efficacy of the Classic UNet may be attributed to its enhanced architectural capacity—most notably, the integration of max-pooling operations and increased encoder-decoder depth—which facilitates the capture of multi-scale spatial features integral to flood dynamics.

Empirical results revealed that the Classic UNet achieved a median Relative Absolute Error (RAE) of approximately 0.11 on the big_test dataset, underscoring its ability to generalize effectively to previously unseen simulations at the patch level. Within this context, the model also performed robustly on prj_01_test, a manually configured subset, achieving a median RAE of 0.11. Given that big_test predominantly comprises less curated simulations from Projects 02, 03, and 04, the observed consistency across datasets indicates that diminished manual preprocessing does not adversely affect generalization to rigorously curated test cases. Such findings affirm the suitability of employing less curated data in future model development strategies.

The closure model, designed to integrate patch-level predictions across entire simulation domains, exhibited elevated RAEs during early temporal intervals. This phenomenon is plausibly attributable to the limited contextual information available when the domain is characterized by initial dryness. Furthermore, the training dataset was disproportionately comprised of patches representing wet conditions or ongoing water accumulation, with relatively few instances capturing transitions from dry to wet states. This inherent temporal bias constrained the model's capacity to generalize to early-stage inundation events. Nevertheless, at later time steps, the closure model surpassed the dummy baseline by a factor of two. Crucially, analyses indicated no discernible correlation between domain size and RAE, thus suggesting that the closure model possesses significant scalability and is amenable to deployment across domains of varying spatial extent.

Several limitations observed in this investigation delineate avenues for subsequent research. First, the study did not assess error propagation in rollout predictions, a consideration essential to understanding the reliability of long-term flood forecasting. Second, neither the deep learning models nor the closure models incorporated physics-based loss functions or stopping criteria that might further enhance predictive accuracy through physical constraint enforcement, these have been shown potential to improve the results (Geltman et al., 2024; Kendler et al., 2022). Third, model training and hyperparameter selection were conducted using a greedy rather than a systematic optimization framework. Additionally, factors pertinent to computational efficiency, such as inference time and model size, were not integrated into the architectural selection process, despite their relevance for operational implementation. Finally, the impact of patch size on prediction accuracy was not systematically evaluated; optimizing patch size could improve the balance between the number of training samples and model precision. Larger patches reduce the prevalence of unknown pixels within the domain (representing internal boundary conditions), which may in turn enhance predictive performance. Addressing these limitations is projected to further advance the practical applicability of deep learning in real-world flood forecasting contexts.

5 Conclusions

This study presents the significant potential of deep learning techniques to emulate hydrodynamic flood simulations. A classic UNet architecture, trained at the patch level, demonstrated strong generalization capabilities across a range of terrains, boundary conditions, and temporal scenarios, surpassing other models in performance. Although the closure model expanded applicability to larger domains, its effectiveness was reduced during early time steps, underscoring ongoing challenges with initializing dry conditions and managing internal boundaries. Nonetheless, observed improvements at later simulation stages and reliable performance over varying domain sizes suggest promising scalability. Further research should prioritize enhancing early-stage prediction accuracy, incorporating physics-based constraints, and refining training methodologies to support practical implementation in real-world flood forecasting.

Acknowledgments

This research was partially funded by the Israeli ministry of science and technology, grants #4348, #5543, and #6796; and the Israeli water authority.

Author Contributions

Lana Khalifa: Methodology, Software, Validation, Formal Validation, Investigation, Data Curation, Writing - Original Draft, Visualization. **Barak Fishbain:** Conceptualization, Methodology, Writing - Review & Editing, Supervision, Funding acquisition.

Software and/or data availability

Name of software and dataset	Deep Learning Flood Modeling
Developer and contact information	Lana Khalifa, lana.khalifa@campus.technion.ac.il
Year first available	2025
Hardware required	Standard PC
Software required	Conda, Python Ver 3.2 and up
Program language	Python
GitHub Link	https://github.com/LanaKhalifa/deep-learning-flood-modeling/tree/main

References

Alhada-Lahbabi, K., Deleruyelle, D., & Gautier, B. (2023). Machine Learning Surrogate Model for Acceleration of Ferroelectric Phase-Field Modeling. *ACS Applied Electronic Materials*, *5*(7), 3894–3907. https://doi.org/10.1021/acsaelm.3c00601

Ang, E. H. W., Wang, G., & Ng, B. F. (2023). Physics-Informed Neural Networks for Low Reynolds Number Flows over Cylinder. *Energies*, *16*(12), Article 12. https://doi.org/10.3390/en16124558

Bermúdez, M., Cea, L., & Puertas, J. (2019). A rapid flood inundation model for hazard mapping based on least squares support vector machine regression. *Journal of Flood Risk Management*, *12*(S1), e12522. https://doi.org/10.1111/jfr3.12522

Chang, L.-C., Shen, H.-Y., Wang, Y.-F., Huang, J.-Y., & Lin, Y.-T. (2010). Clustering-based hybrid inundation model for forecasting flood inundation depths. *Journal of Hydrology*, *385*(1), 257–268. https://doi.org/10.1016/j.jhydrol.2010.02.028

Chen, Y., Dodwell, T., Chuaqui, T., & Butler, R. (2023). Full-field prediction of stress and fracture patterns in composites using deep learning and self-attention. *Engineering Fracture Mechanics*, *286*, 109314. https://doi.org/10.1016/j.engfracmech.2023.109314

Cheng, C., & Zhang, G.-T. (2021). Deep Learning Method Based on Physics Informed Neural Network with Resnet Block for Solving Fluid Flow Problems. *Water*, *13*(4), Article 4. https://doi.org/10.3390/w13040423

Elkhrachy, I. (2022). Flash Flood Water Depth Estimation Using SAR Images, Digital Elevation Models, and Machine Learning Algorithms. *Remote Sensing*, *14*(3), Article 3. https://doi.org/10.3390/rs14030440

Food and Agriculture Organization, United Nations. (2025, August 10). *Rivers of North America—"FAO catalog"*. https://data.apps.fao.org/catalog/dataset/9264483f-ca14-496b-aeae-fe1b8aebf520

Geltman, A., Levy, I., & Fishbain, B. (2024). Machine Education Approach for Generating Accurate NO_2 and PM_(2.5) Dense Pollution Maps in Israel. *Environmental Science & Technology - Air*.

Guo, Z., Leitão, J. P., Simões, N. E., & Moosavi, V. (2021). Data-driven flood emulation: Speeding up urban flood predictions by deep convolutional neural networks. *Journal of Flood Risk Management*, *14*(1), e12684. https://doi.org/10.1111/jfr3.12684

Guo, Z., Moosavi, V., & Leitão, J. P. (2022). Data-driven rapid flood prediction mapping with catchment generalizability. *Journal of Hydrology*, *609*, 127726. https://doi.org/10.1016/j.jhydrol.2022.127726

Hosseiny, H., Nazari, F., Smith, V., & Nataraj, C. (2020). A Framework for Modeling Flood Depth Using a Hybrid of Hydraulics and Machine Learning. *Scientific Reports*, *10*(1), 8222. https://doi.org/10.1038/s41598-020-65232-5

Hydrologic Engineering Center. (2021). *HEC-RAS User's Manual, U.S. Army Corps of Engineers, Davis CA., April 2021.* U.S. Army Corps of Engineers,.

Kabir, S., Patidar, S., Xia, X., Liang, Q., Neal, J., & Pender, G. (2020). A deep convolutional neural network model for rapid prediction of fluvial flood inundation. *Journal of Hydrology*, *590*, 125481. https://doi.org/10.1016/j.jhydrol.2020.125481

Kendler, S., Mano, Z., Aharoni, R., Raich, R., & Fishbain, B. (2022). Hyperspectral imaging for chemicals identification: A human-inspired machine learning approach. *Scientific Reports 2022 12:1*, *12*(1), 1–10. https://doi.org/10.1038/s41598-022-22468-7

Löwe, R., Böhm, J., Jensen, D. G., Leandro, J., & Rasmussen, S. H. (2021). U-FLOOD – Topographic deep learning for predicting urban pluvial flood water depth. *Journal of Hydrology*, *603*, 126898. https://doi.org/10.1016/j.jhydrol.2021.126898

Mosavi, A., Ozturk, P., & Chau, K. (2018). Flood Prediction Using Machine Learning Models: Literature Review. *Water*, *10*(11), Article 11. https://doi.org/10.3390/w10111536

National Oceanic and Atmospheric Administration. (2025, August 10). *Flooded Locations And Simulated Hydrographs (FLASH) progect*. Flooded Locations And Simulated Hydrographs (FLASH) Progect. https://inside.nssl.noaa.gov/flash/database/database-2016v1/

Nguyen, P. C. H., Nguyen, Y.-T., Seshadri, P. K., Choi, J. B., Udaykumar, H. S., & Baek, S. (2023). A Physics-Aware Deep Learning Model for Energy Localization in Multiscale Shock-To-Detonation Simulations of Heterogeneous Energetic Materials. *Propellants, Explosives, Pyrotechnics*, *48*(4), e202200268. https://doi.org/10.1002/prep.202200268

Obiols-Sales, O., Vishnu, A., Malaya, N., & Chandramowliswharan, A. (2020). CFDNet: A deep learning-based accelerator for fluid simulations. *Proceedings of the 34th ACM International Conference on Supercomputing*, 1–12. https://doi.org/10.1145/3392717.3392772

Pan, T.-Y., Lai, J.-S., Chang, T.-J., Chang, H.-K., Chang, K.-C., & Tan, Y.-C. (2011). Hybrid neural networks in rainfall-inundation forecasting based on a synthetic potential inundation database. *Natural Hazards and Earth System Sciences*, *11*(3), 771–787. https://doi.org/10.5194/nhess-11-771-2011

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In N. Navab, J. Hornegger, W. M. Wells, & A. F. Frangi (Eds.), *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015* (pp. 234–241). Springer International Publishing. https://doi.org/10.1007/978-3-319-24574-4_28

Santos, J. E., Xu, D., Jo, H., Landry, C. J., Prodanović, M., & Pyrcz, M. J. (2020). PoreFlow-Net: A 3D convolutional neural network to predict fluid flow through porous media. *Advances in Water Resources*, *138*, 103539. https://doi.org/10.1016/j.advwatres.2020.103539

Teng, J., Jakeman, A. J., Vaze, J., Croke, B. F. W., Dutta, D., & Kim, S. (2017). Flood inundation modelling: A review of methods, recent advances and uncertainty analysis. *Environmental Modelling & Software*, *90*, 201–216. https://doi.org/10.1016/j.envsoft.2017.01.006

US EPA, O. (2015, November 25). *Ecoregions of North America* [Data and Tools]. https://www.epa.gov/eco-research/ecoregions-north-america

US GEological Survey. (2025, October 8). *3D elevation program (3DEP)*. 3D Elevation Program (3DEP). https://www.usgs.gov/3d-elevation-program