Deep Learning-Based Prediction of Coastal Storm Tide Flood Maps

Molly McKenna
Advisor: J.C. Dietrich

Acknowledgements

- ★ Advisor: Casey Dietrich
- ★ Committee: Katherine Anarde, John Baugh
- ★ Tomás Cuevas Lopez and Dylan Anderson
- ★ Everyone in the Coastal Lab Team
- ★ My family, friends and community in Raleigh
- ★ My cat Peanut Butter

Without all of you this would have been so much harder, thank you

About Me...

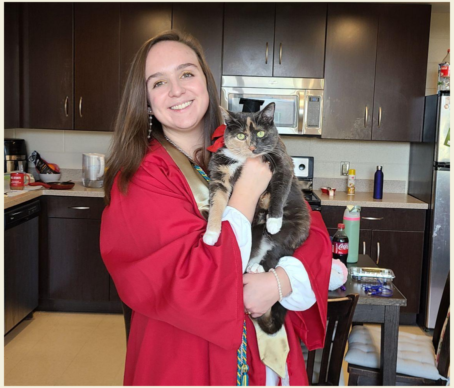
NC born and raised...

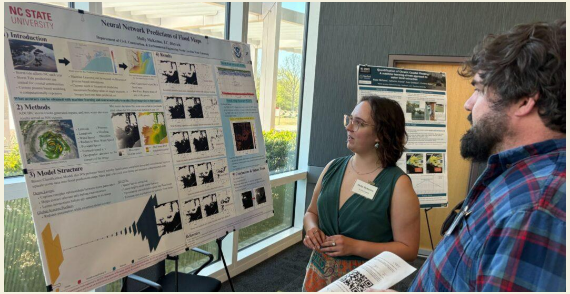
Undergraduate degree from NC State in Civil Engineering with a minor in Computer Programming (almost got my degree in comp sci)

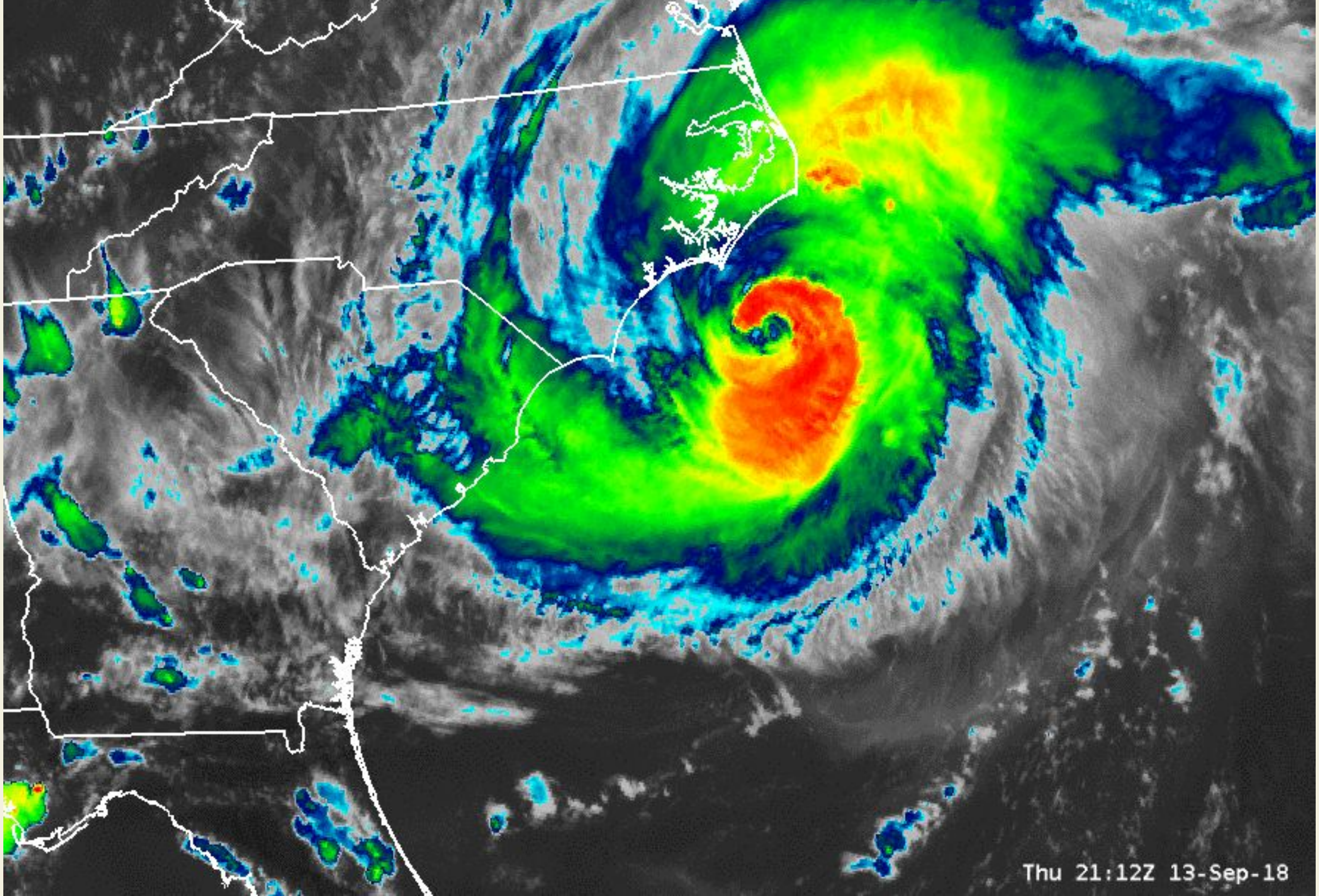
But I fell in love with coastal engineering and research...

In my senior year of college I worked with Casey in undergraduate research

Started my masters degree in coastal engineering the next fall.







NWS

Hurricane Florence 2018


Wilmington, NC


Atlantic Beach, NC

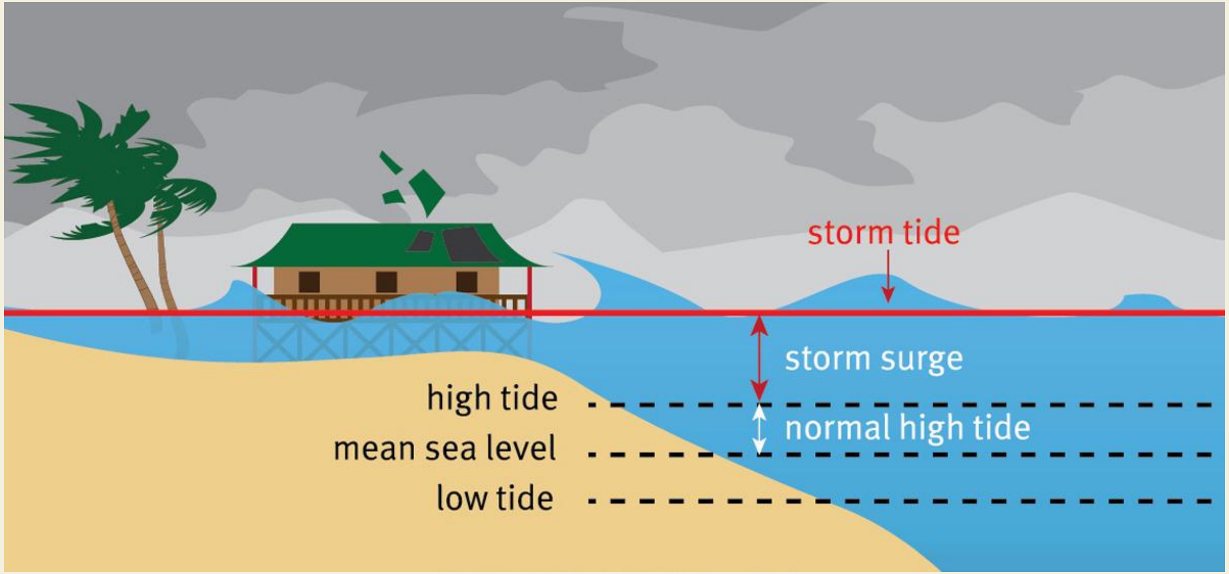

Atlantic Beach, NC


North Topsail Beach, NC

But what is Storm Tide?

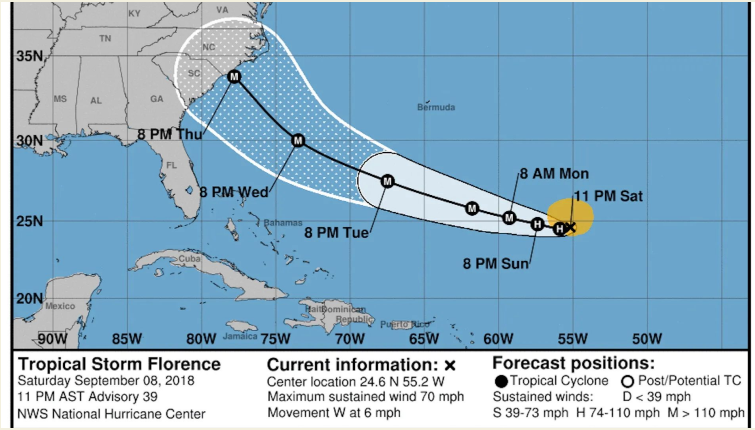
Storm Tide is the combined effect of storm surge along with tide.

This nonlinear interaction between storm surge and tide are hard for models to generalize.



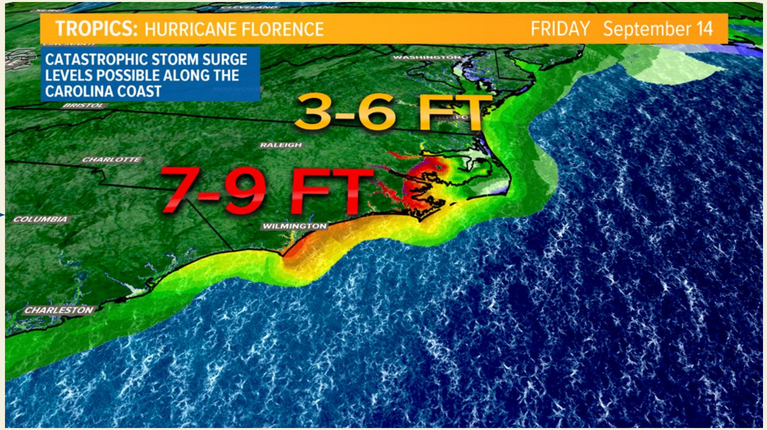
NOAA storm tide informational graphic

How We Currently Predict Storm Tide...



NHC Predicted Track:
Every 6 hours, a new predicted track is released

Storm Model



Storm Surge Prediction and Advisory

But what is the model?

Storm Flood Models

Process Based

Surrogate

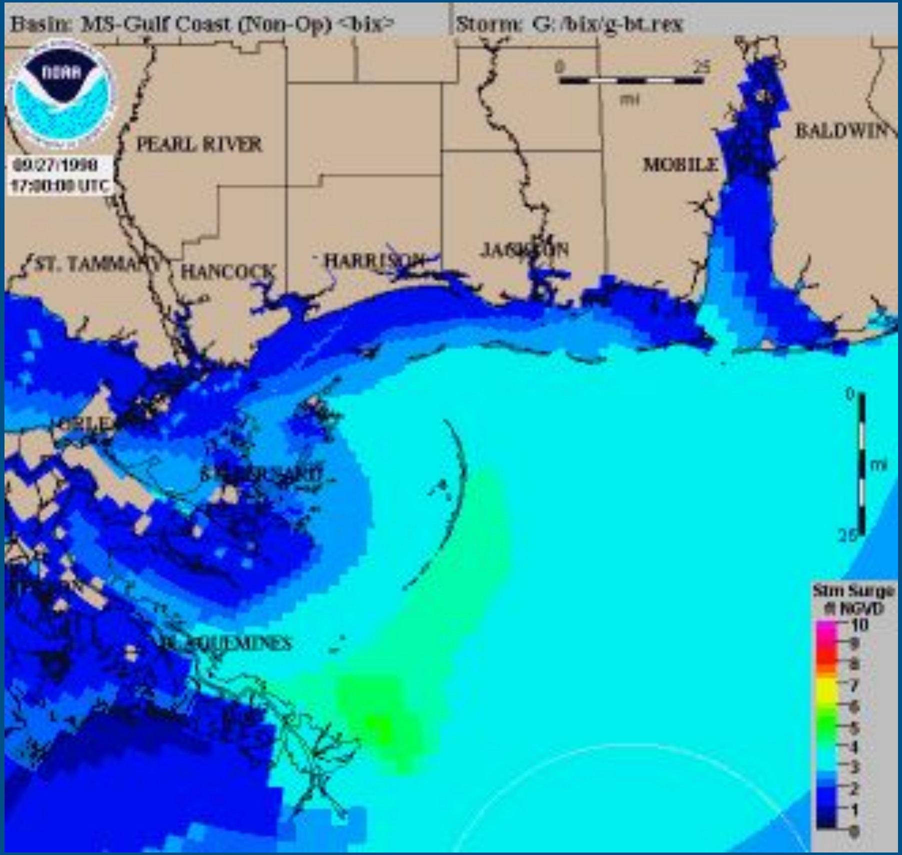
Process Based Models

Process Based models:

- Numerical models that simulate physical processes
- Solve the governing equations, not statistical patterns
- Used to predict storm surge and coastal flooding

Process based models for coastal flooding:

- ADCIRC, SLOSH, SFINCS, Delft3D



Development of a hurricane by SLOSH model run [NOAA]

ADCIRC

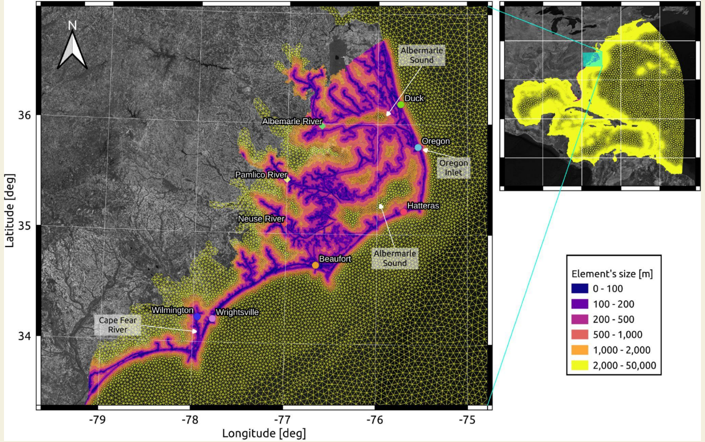
Solves the depth-integrated shallow-water equations

Unstructured finite-element mesh

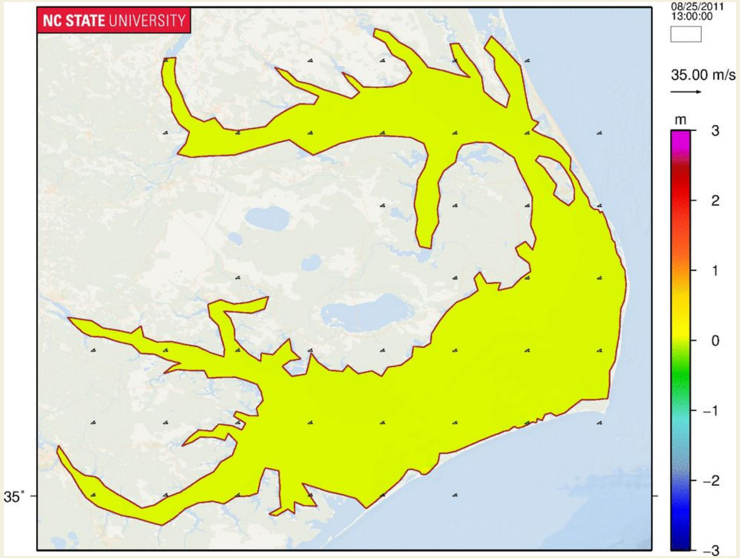
Very high spatial resolution in complex coastal areas

Accurate, physics-consistent coastal dynamics

Runtime: hours to days on HPCs



Example ADCIRC Mesh [Cuevas Lopez et al. 2025]



Example ADCIRC model run: with water level outputs [CCHT]

Storm Surge Models

Process Based

Surrogate

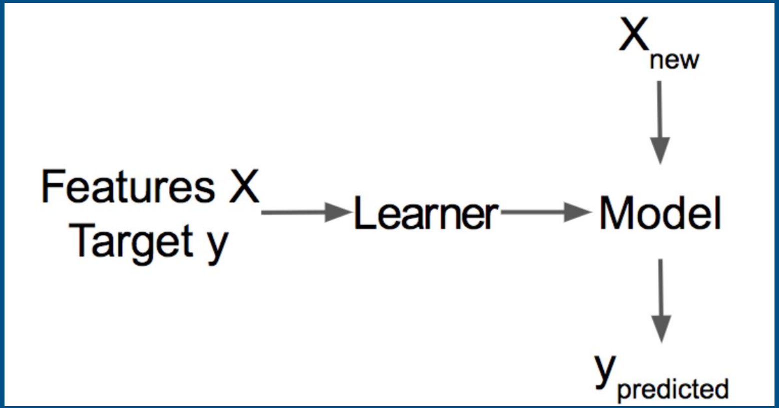
What are Surrogate Models?

Learn relationships between inputs and outputs from data

ML models trained to approximate expensive simulations

Enable ensembles, uncertainty quantification, rapid updates

Support real-time or near-real-time prediction



Example of a surrogate model workflow

Surrogate Models for Coastal Flooding

1. Early Models: point-based surge predictions
2. Later Models: hybrid models, hundred/thousand point surge predictions
3. Storm Surge predictions of coarse spatial maps
4. First storm-tide Surrogate model: point-based

Early Surrogate Models for Storm Surge

First models predicted surge at a handful of coastal locations.

Predicted surge at a small number of coastal locations

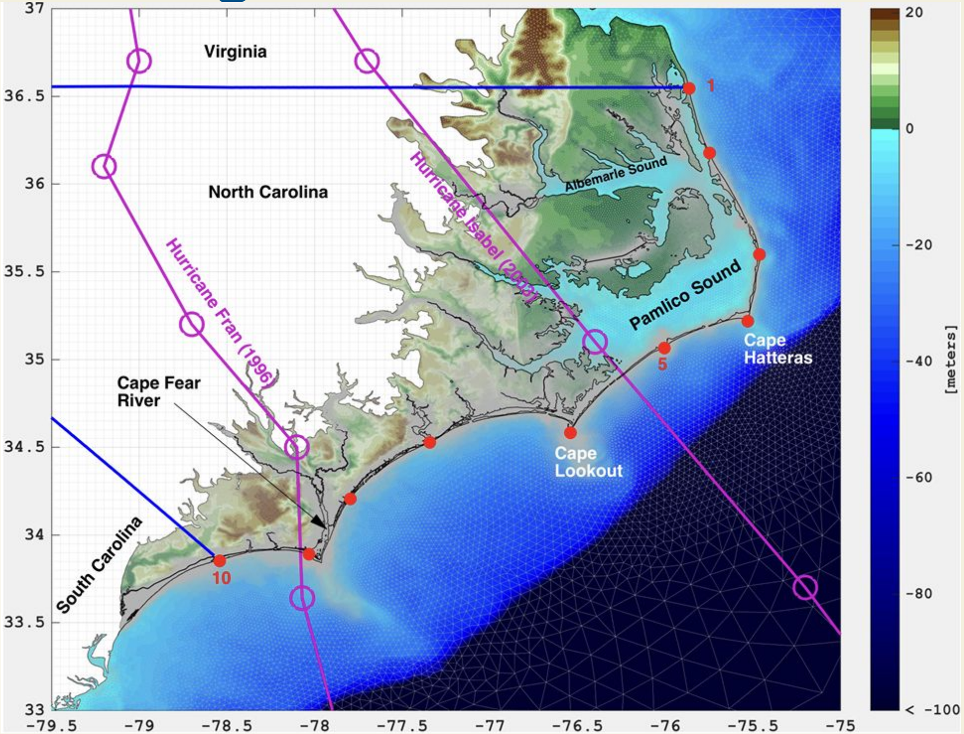
Inputs: storm parameters (track, intensity, pressure)

Fast and computationally cheap

No spatial information — point-wise only

Useful for quick estimates, not mapping

Examples: Bezuglov et al. (2016), Hashemi et al. (2016)



ADCIRC mesh of NC. Ten output locations as red points. The paths of Hurricane Fran and Isabel are shown. (Bezuglov et al., 2016, Figure 1)

Hybrid Model for Storm Surge

Models Scale up to thousands of points and incorporate more physics-relevant inputs.

Scaled up to hundreds–thousands of coastal points

Expanded inputs: storm track time series, wind fields, pressure fields

Used more advanced ML/NN architectures
- Deep autoencoders
- Deep neural networks
- Multistage classification–regression models

Still discrete predictions, not continuous maps

Predicted storm surge only, not total water level

Examples: Saviz Naeini & Snaiki (2024), Pachev et al. (2023)

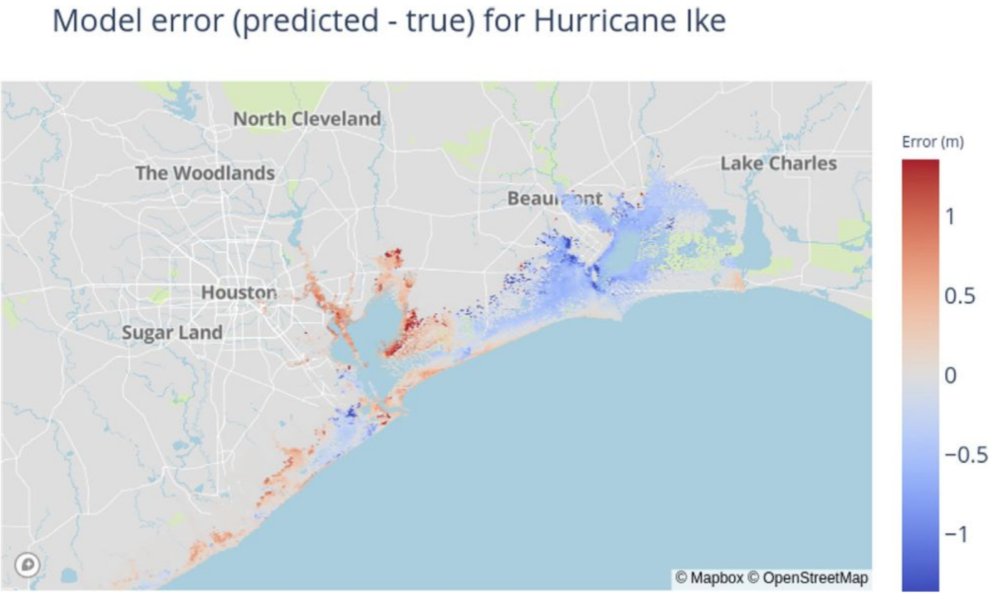


Fig. 10. Model error for Ike. Map data courtesy of OpenStreetMap (OpenStreetMap contributors, 2022).

Model Error of the Neural Network model in TX for Hurricane Ike [Pachev et al. 2023]

Spatial ML for Storm Surge

Predicted gridded floods, but at low resolution.

CNNs used to predict coarse flood maps (ex 128 x 128 pixels over 2.5◦ x 2.5◦)

Pixel resolution: ~2km x ~2km

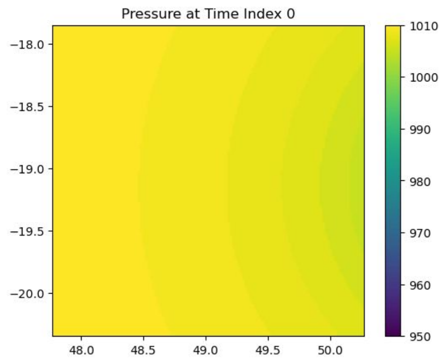
Land–water mask restricts predictions to “wet” areas

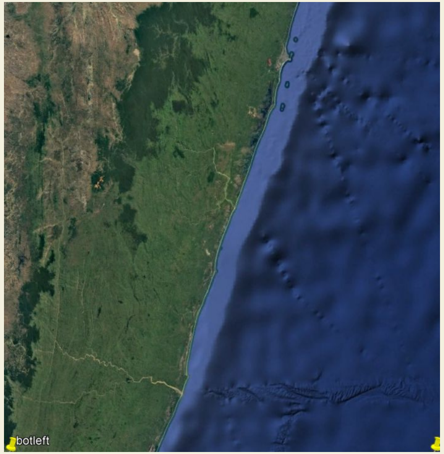
- No floodplains or overland flooding

Cannot resolve estuaries, channels, barrier islands

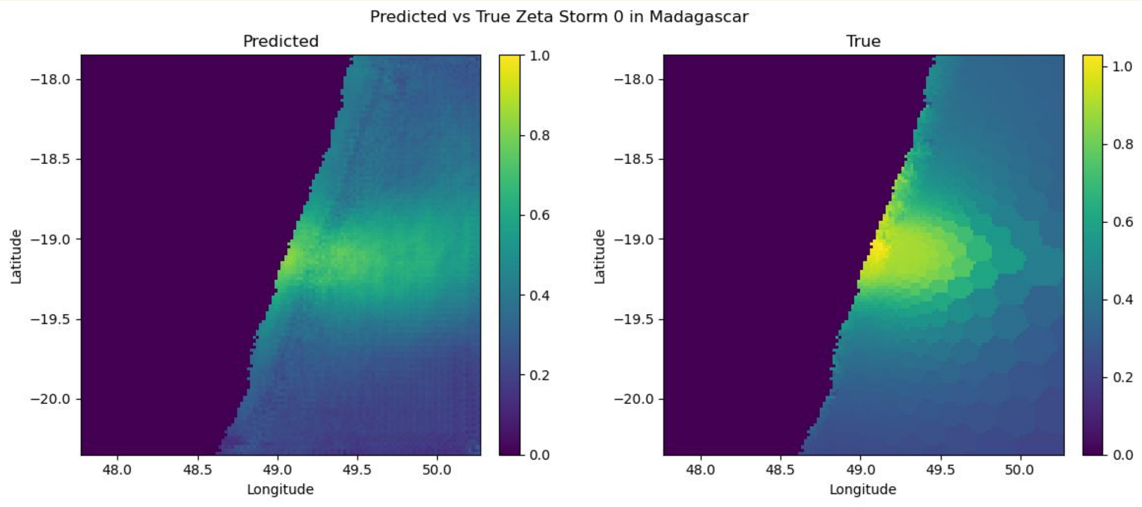
Surge-Only Model

Example dataset: NSF Data Depot (Pachev, 2026)





Eastern coast of Madagascar



Point-Based ML for Storm Tide

First ML model to predict storm tide. (Cuevas Lopez 2025)

Predicts storm tide not surge.

Inputs include:

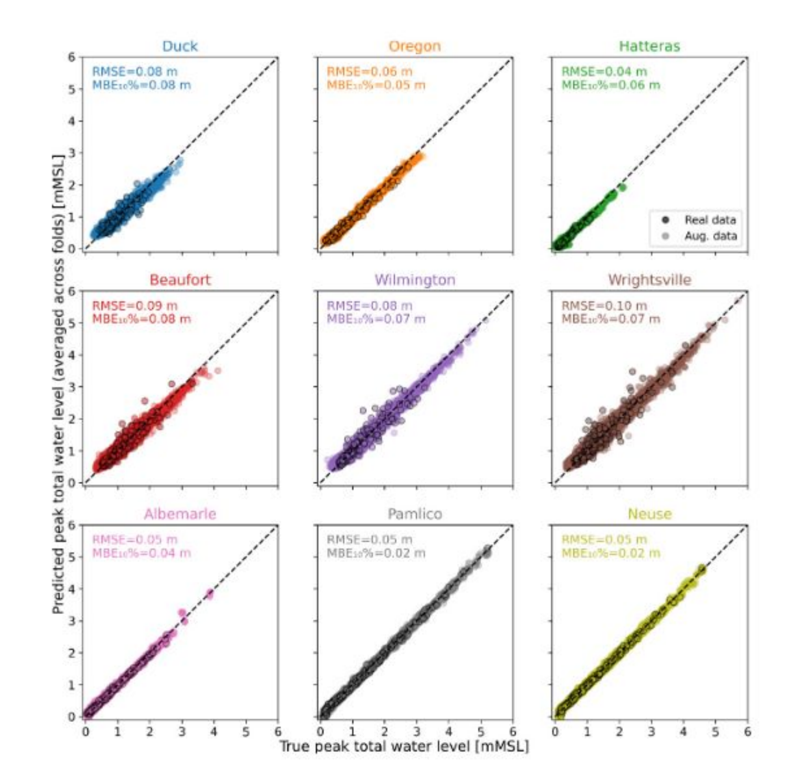
- Storm track
- Storm evolution
- Tidal phase

Outputs: peak storm tide at 9 coastal locations

Highly accurate: 0.09 - 0.19 m error

Captures nonlinear tide-surge interaction

Still point-based, not spatial



Although these studies show that neural networks can generalize coastal processes, currently, none of these models are using NN/ML to predict **storm tide maps.**

How can neural network models learn the relationships between tides, coastal storms, and coastal landscapes to predict maps of storm tide flooding?

A convolutional neural network (CNN), whose architecture is suited to learning hierarchical and coherent patterns,

CNNs are built to learn spatial patterns and multi-scale structure.

can extract the physical structure of storm-tide flooding from a large library of high-resolution simulations and generate accurate flood maps at forecast timescales.

CNNs learn physical storm–flood relationships from many simulations and can predict flood maps quickly.

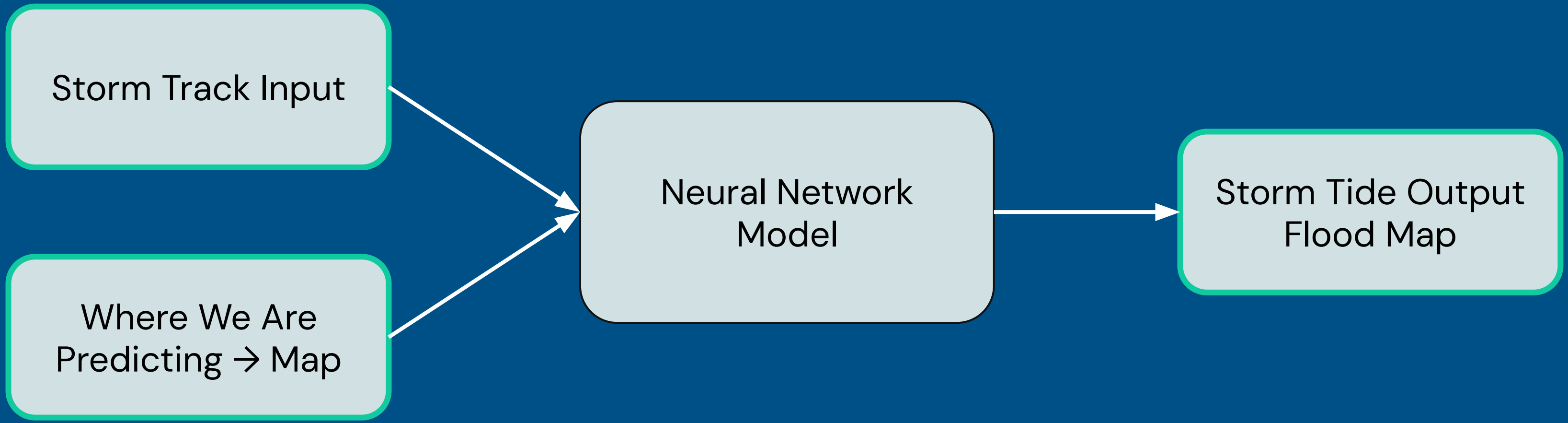
By encoding storm-scale forcing and high-resolution coastal morphology in a continuous representation,

We encode storms and coastal morphology as continuous, high-resolution images...

the architecture is tailored to the physics of coastal flooding—capturing multi-scale interactions, flood-propagation pathways, and the land-water connectivity that governs storm-tide response.

The model learns how storms interact with complex coastal features to produce storm-tide flooding.

Model Overview



Why Inputs and Outputs? Surrogate models map inputs to outputs.

Model Basis:
(Cuevas Lopez 2025)

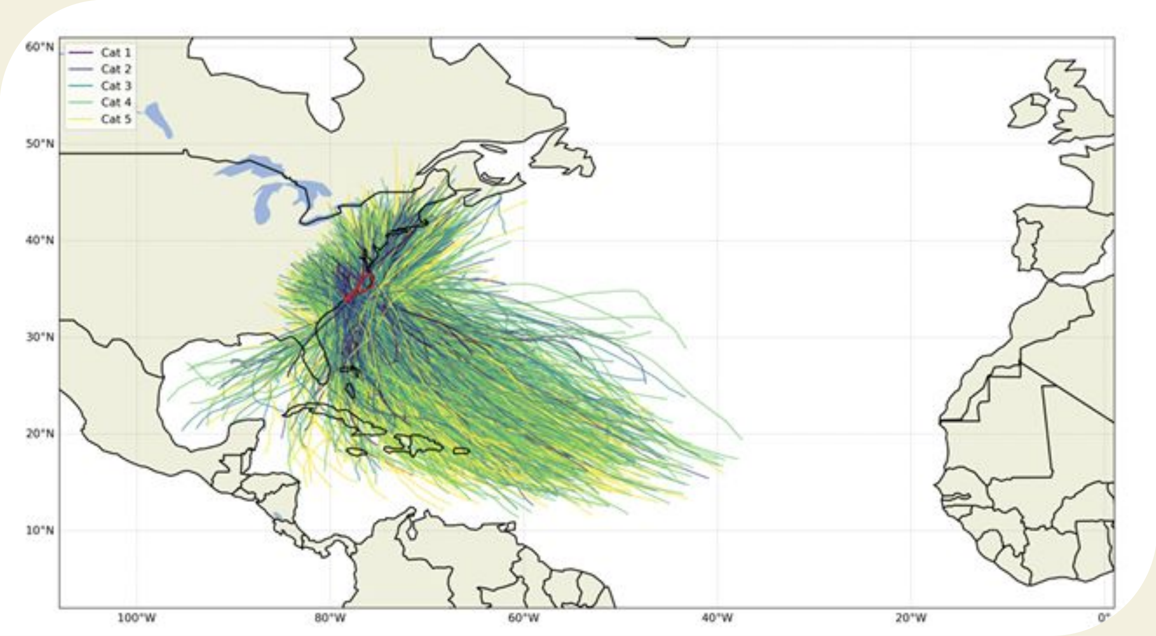
1,813 synthetic Atlantic storms selected (From STORM, a statistical extension of historical storms)

Storms were chosen that specifically impacted NC
- Tracks long enough to influence NC
- Storms reaching ≥ Category 1 wind speeds

Each storm run in ADCIRC with simulated tides + atmospheric forcing

These 1,813 simulations took months to run

Produces high-resolution storm-tide simulations for training

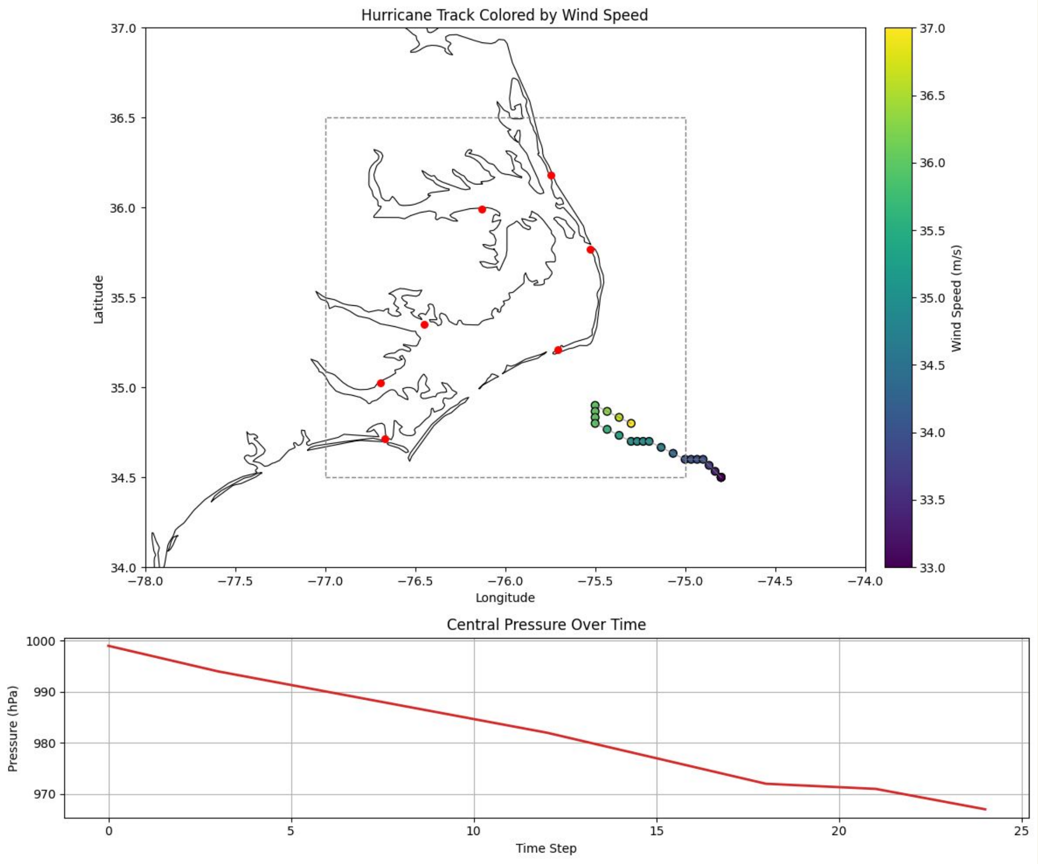


Selection of Storms that impact NC [Cuevas Lopez et al. 2025]

Selected Track Inputs

This is a time series of the most important 25 hours.

- Storm longitude and latitude
- Wind Speed
- Pressure
- Radius to Max Wind Speed
- Heading Direction
- Forward Speed, u, v
- Tidal locations (Duck, Hatteras, Beaufort, Albemarle, Pamlico, Neuse, Oregon Inlet)
- Storm dist u,v to tidal location
- Distance to image center
- Distance u, v to image center



Land Input

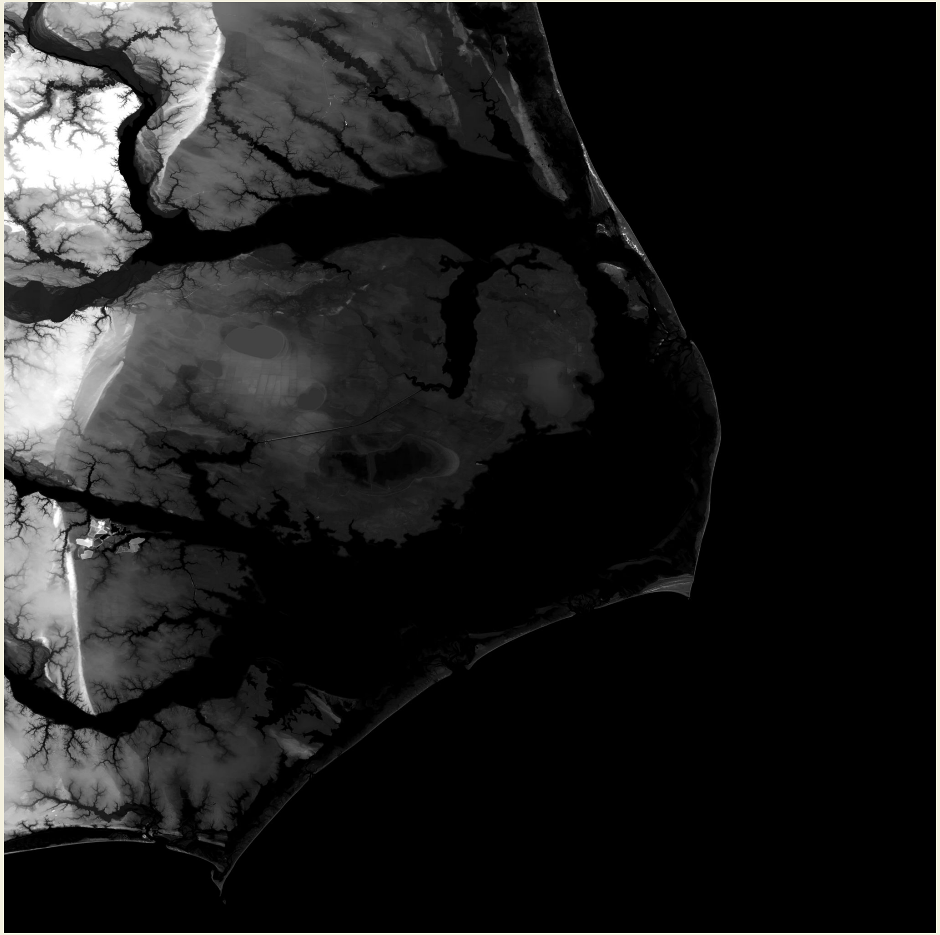
Represents the land and coastal morphology of the prediction region

Same spatial domain as the output storm-tide map

Provides the model with a reference frame for where flooding can occur

Gives the model the initial conditions of the landscape

This helps the model focus on the storm flooding and not the dry land elevations



ADCIRC Outputs

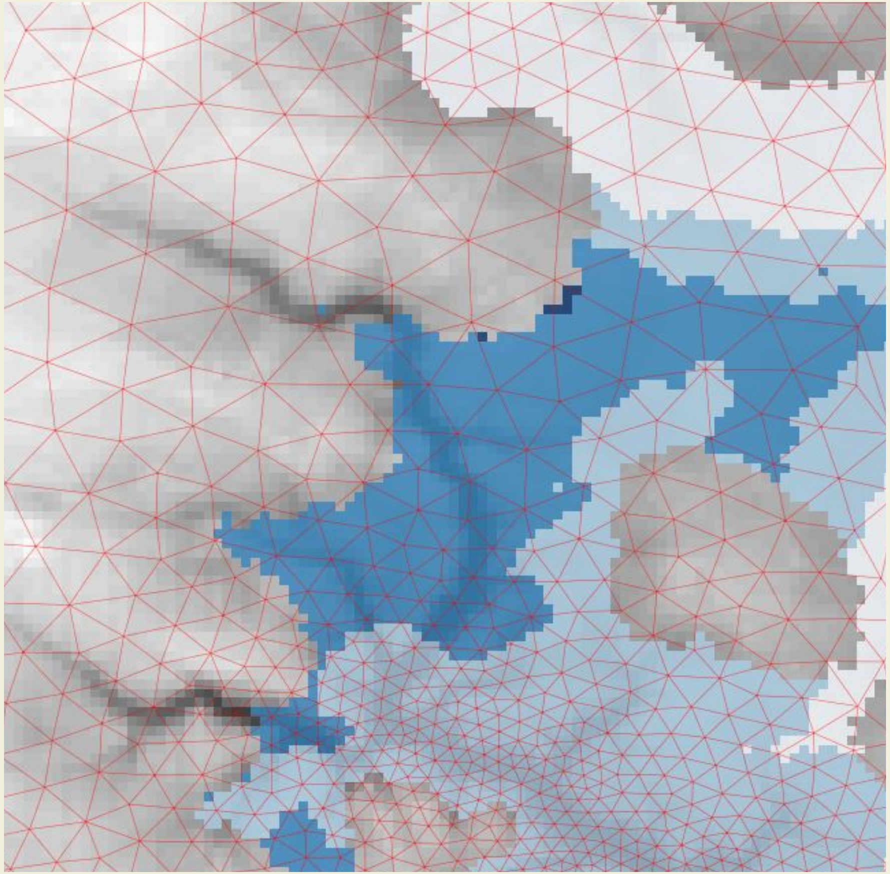
ADCIRC outputs water levels on an unstructured triangular mesh

We extract the maximum water elevation for each storm → storm tide max water elevation for each storm

We interpolate max water elevation onto a regular grid

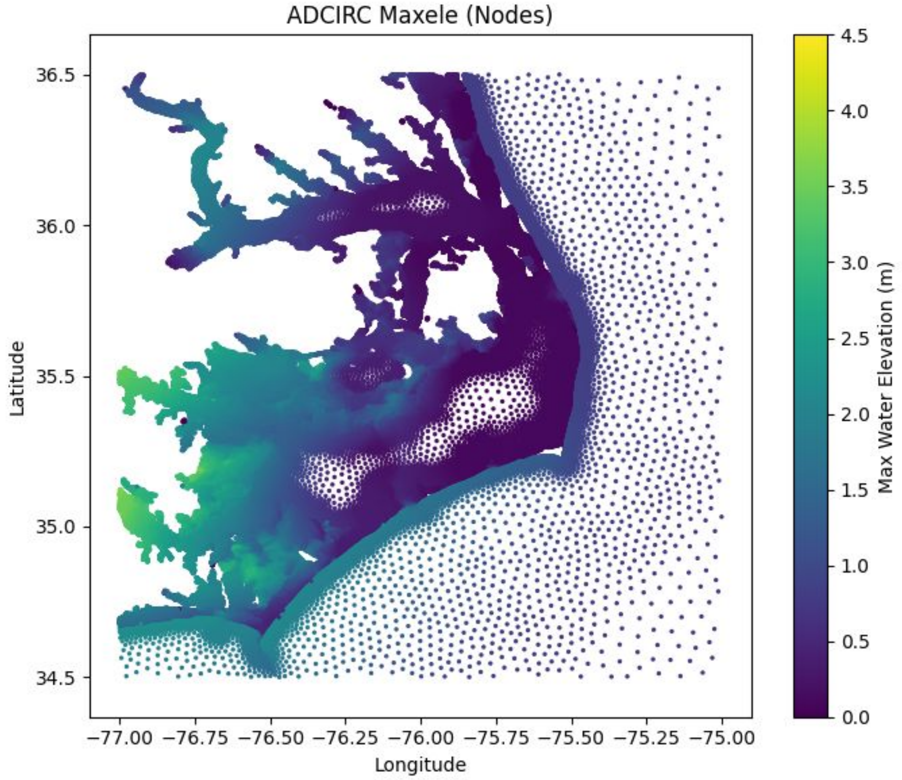
Wet-node values are mapped into DEM pixel locations

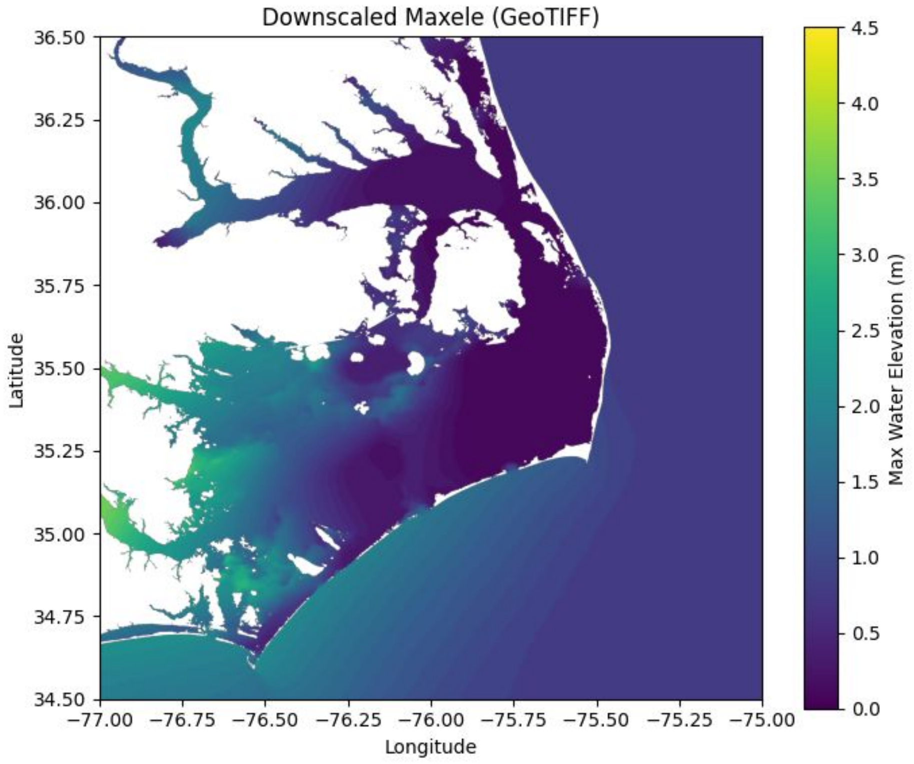
Triangular mesh boundaries become continuous raster surfaces



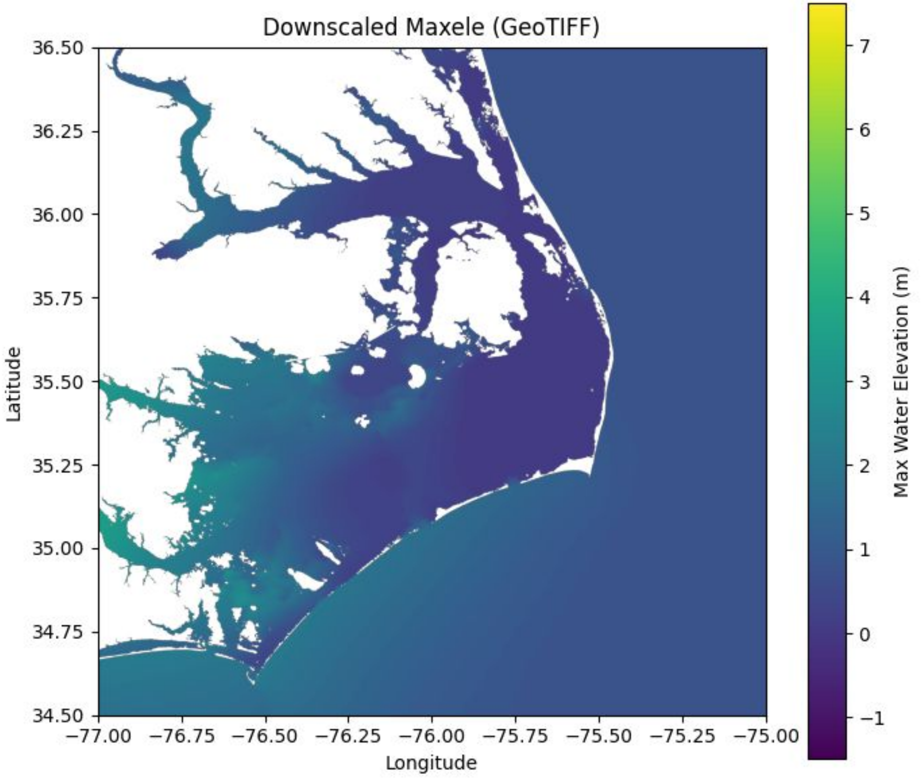
ADCIRC Downscaling flood example:
Red: ADCIRC mesh edges
Blue: Flood values downscaled form ADCIRC.
(this is 0.04x0.04 long lat extent, at the image resolution)

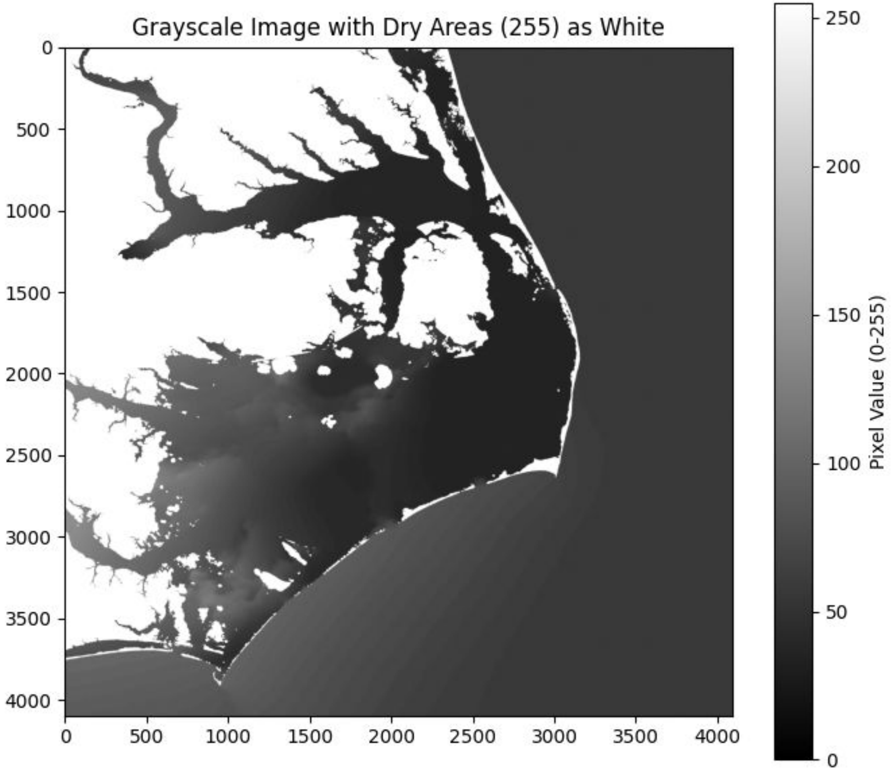
ADCIRC Outputs → Raster Surface



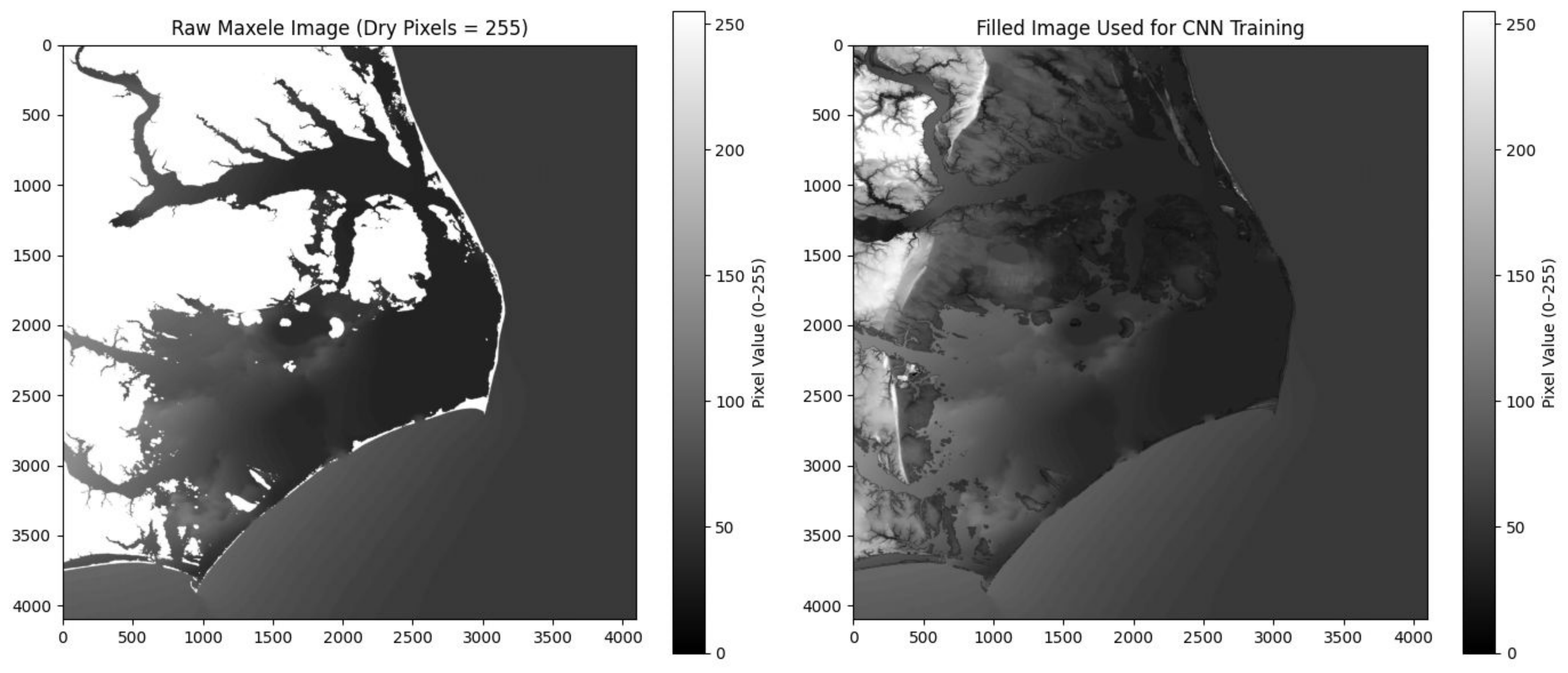


Raster Surface → Grayscale Image





Grayscale Image → Continuous Surface



But 1,813 storm-tide images were not enough for the model to learn how storms impact the entire region...

We also didn’t/couldn’t model more ADCIRC storms because the 1813 took months to compute.

Surrogate models require large, diverse datasets to generalize.

To combat this we augmented the data...

Data Augmentation

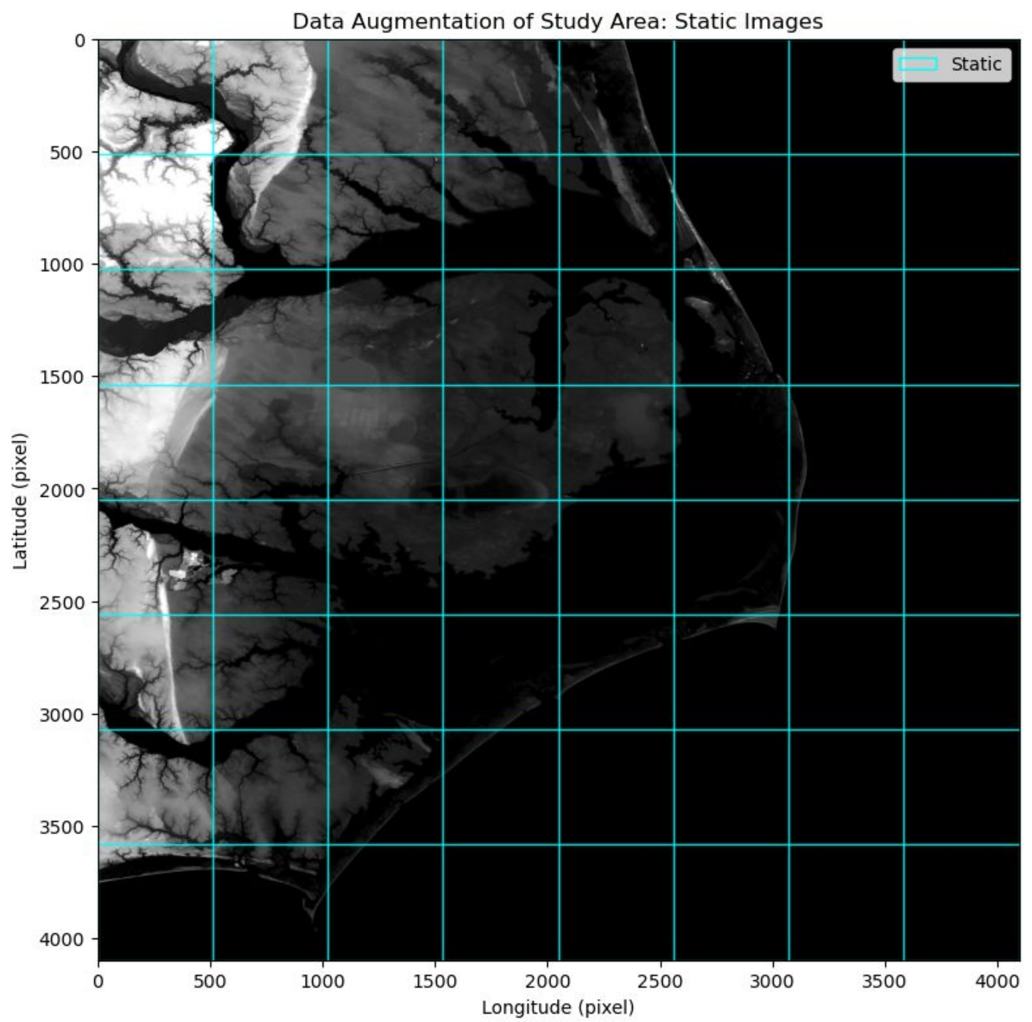
We divide the full 2°×2° domain into 64 fixed centerpoints

Each centerpoint corresponds to a 0.25°×0.25° tile

Every storm is sampled at all centerpoints

Ensures full spatial coverage of the study area

Expands the dataset from 1,813 storms → 116,032 storm-tide tiles



Train/Val/Test Dataset Split

116,032 total samples

Random shuffle applied before splitting

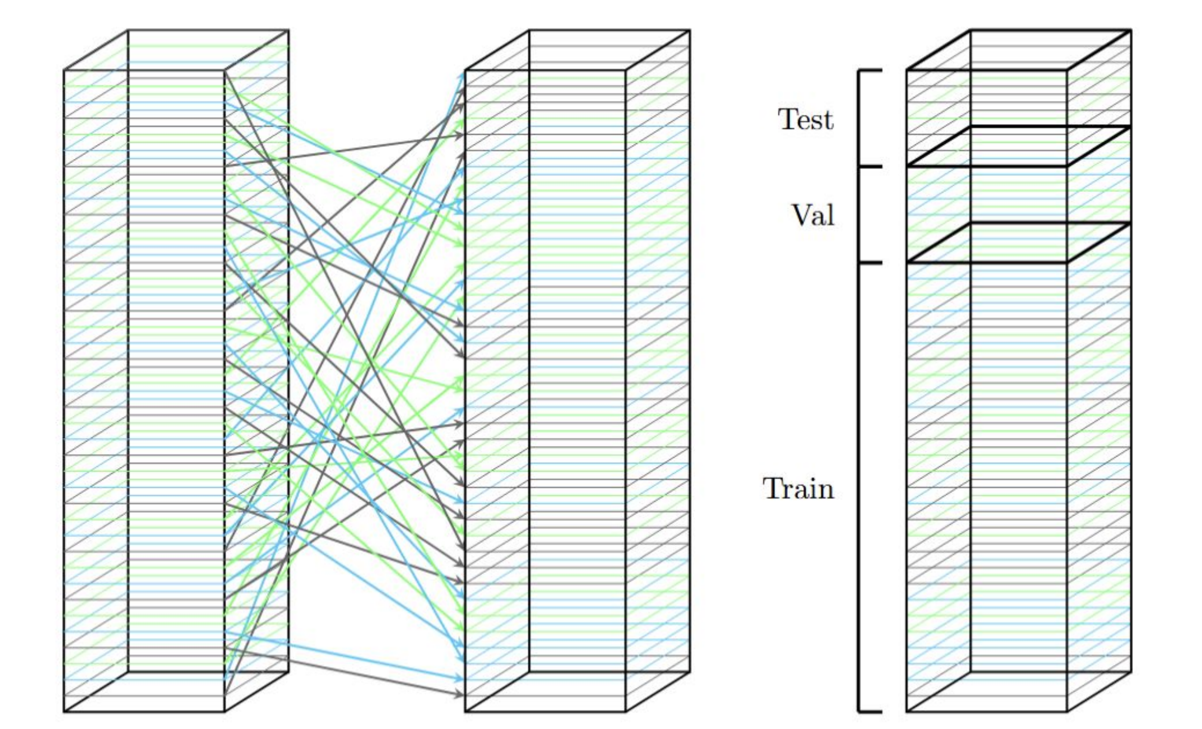
70% → Training set (model learns weights)

15% → Validation set (model tunes hyperparameters, early stopping)

15% → Test set (held-out, never seen during training)

Train + Val are the only samples the model ever learns from

Test set evaluates true generalization on unseen storms/tiles

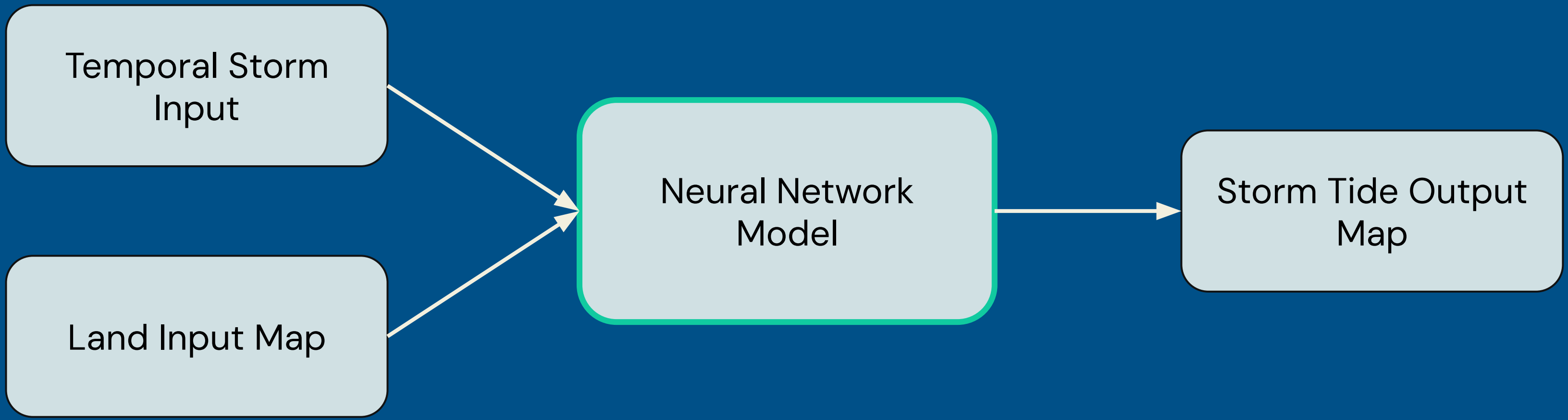


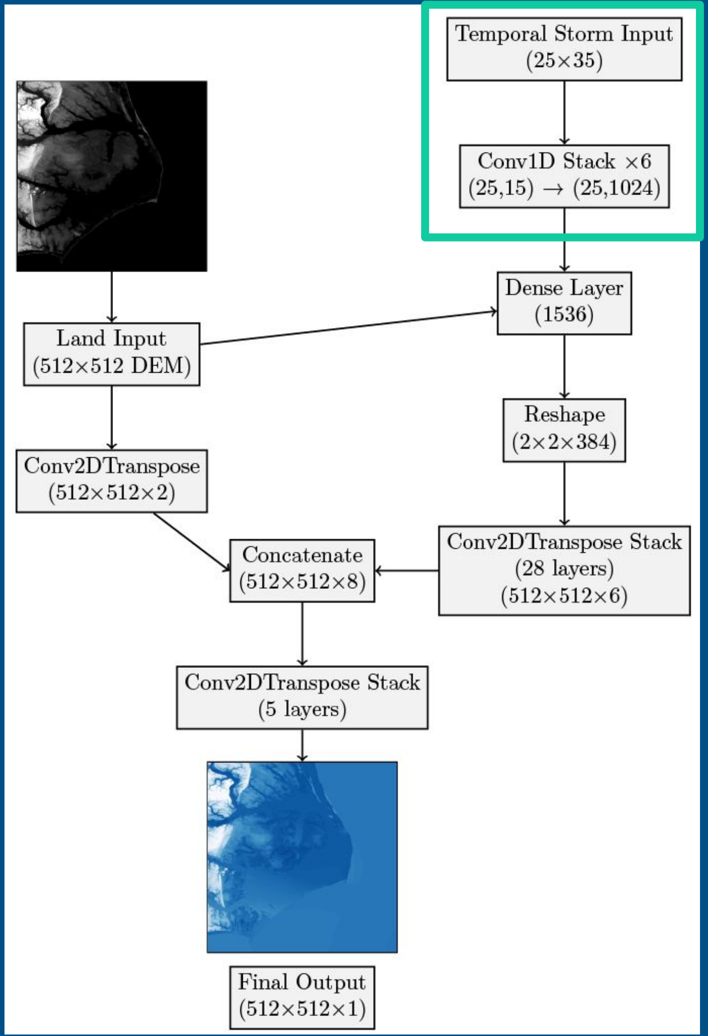
Model Library Summary

- Inputs: Storm intensity + tidal time series + geographic information of the storm to the prediction location.
- Inputs: land map for geospatial reference
- Outputs: Storm tide maps of max flooding for each storm for 0.25∘ longitude and latitude extents.

- **But how would a model relate these storm tracks to these flood images?**

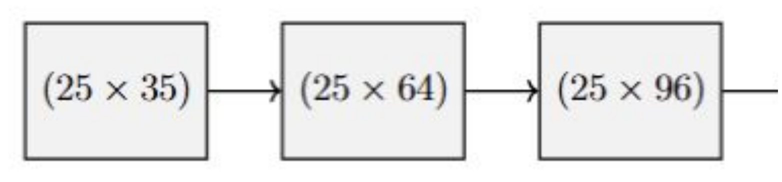
Model Overview




Temporal Storm Input
(25×35)
Conv1D Stack ×6
(25,15) → (25,1024)
Dense Layer
(1536)
Land Input
(512×512 DEM)
Reshape
(2×2×384)
Conv2DTranspose
(512×512×2)
Concatenate
(512×512×8)
Conv2DTranspose Stack
(28 layers)
(512×512×6)
Conv2DTranspose Stack
(5 layers)
Final Output
(512×512×1)

1D CNNs

1D CNNs process the temporal storm-track sequence

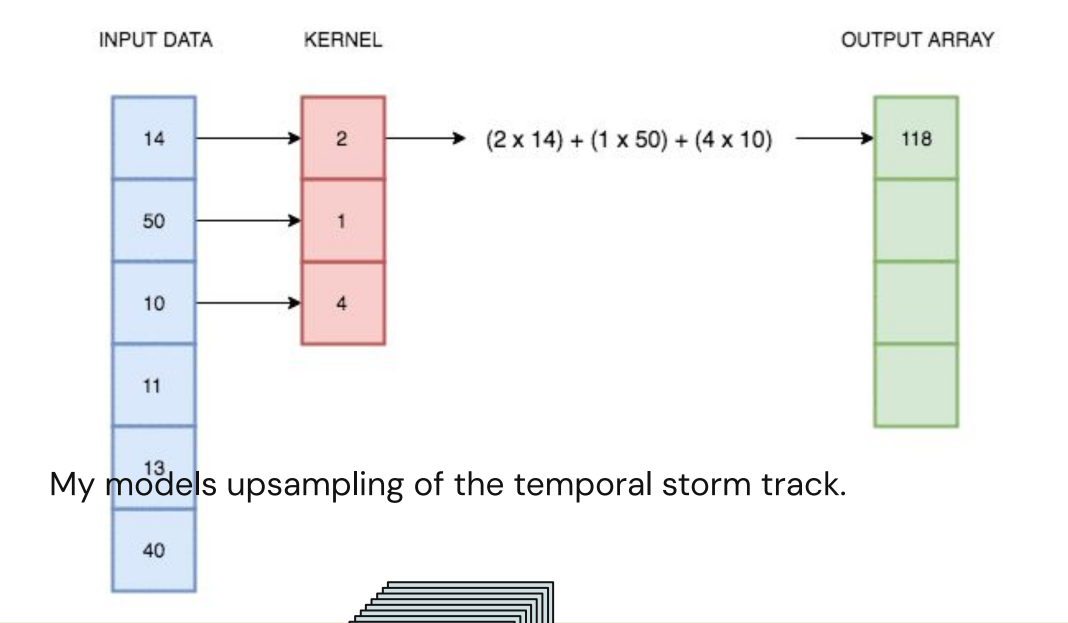


while extracting meaningful patterns

Upsampling layers expand the temporal features for fusion with spatial inputs

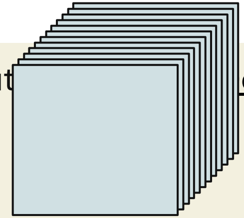
Helps the model learn how storm evolution shapes the final flood map.

The upsampling builds into a 2x2 seed image for the next stage of the model.

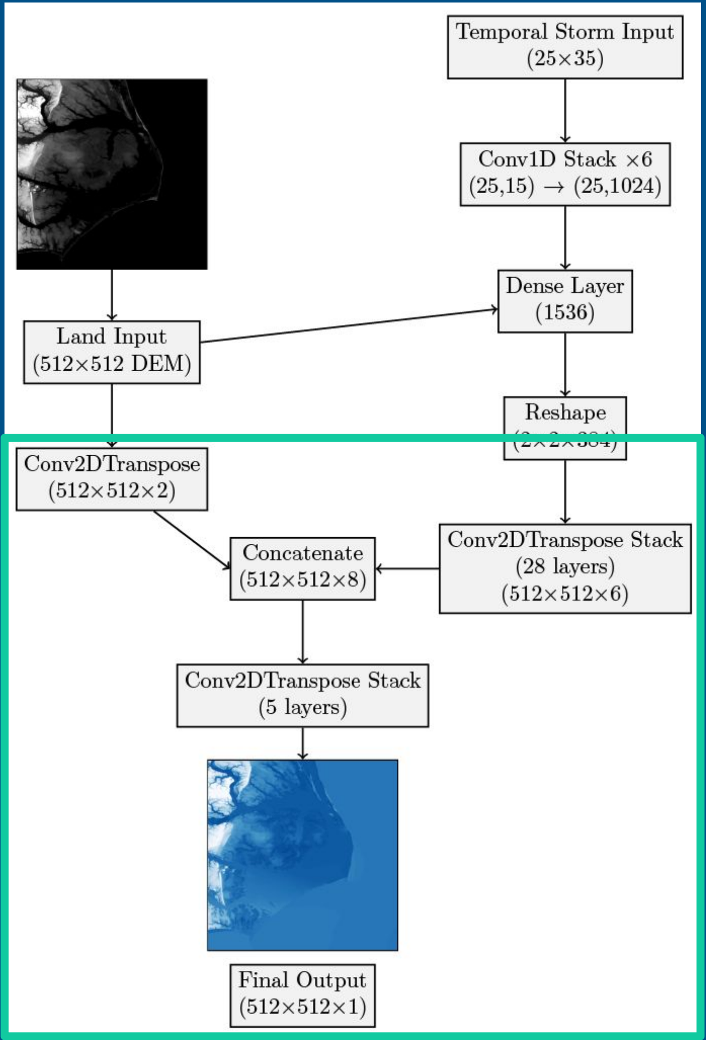


My models upsampling of the temporal storm track.

1D Convolut eep Learning Overview)



Example of the stack of seed images, the input to the next stage is 384, 2x2 images.


Temporal Storm Input
(25×35)
Conv1D Stack ×6
(25,15) → (25,1024)
Dense Layer
(1536)
Reshape
Land Input
(512×512 DEM)
Conv2DTranspose
(512×512×2)
Concatenate
(512×512×8)
Conv2DTranspose Stack
(28 layers)
(512×512×6)
Conv2DTranspose Stack
(5 layers)
Final Output
(512×512×1)

2D Convolutional Layers

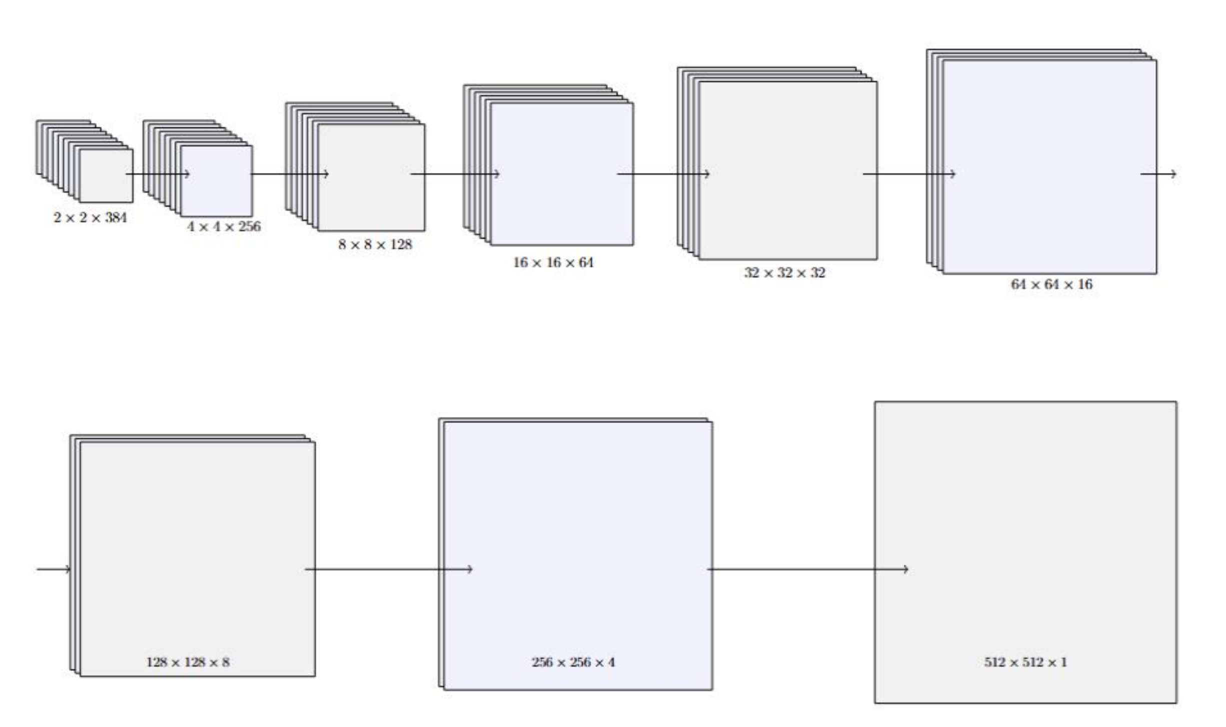
Conv2DTranspose layers progressively upsample the 2×2×384 latent into the full 512×512 image

Each upsampling step increases spatial resolution and adds spatial complexity

Early layers learn coarse, low-resolution flood patterns; later layers refine fine-scale structure

Land elevation is re-introduced at high resolution to anchor the model to the correct geography

This guides the model to focus on the specific tile being predicted



Overall Model Structure

Dual-input architecture:

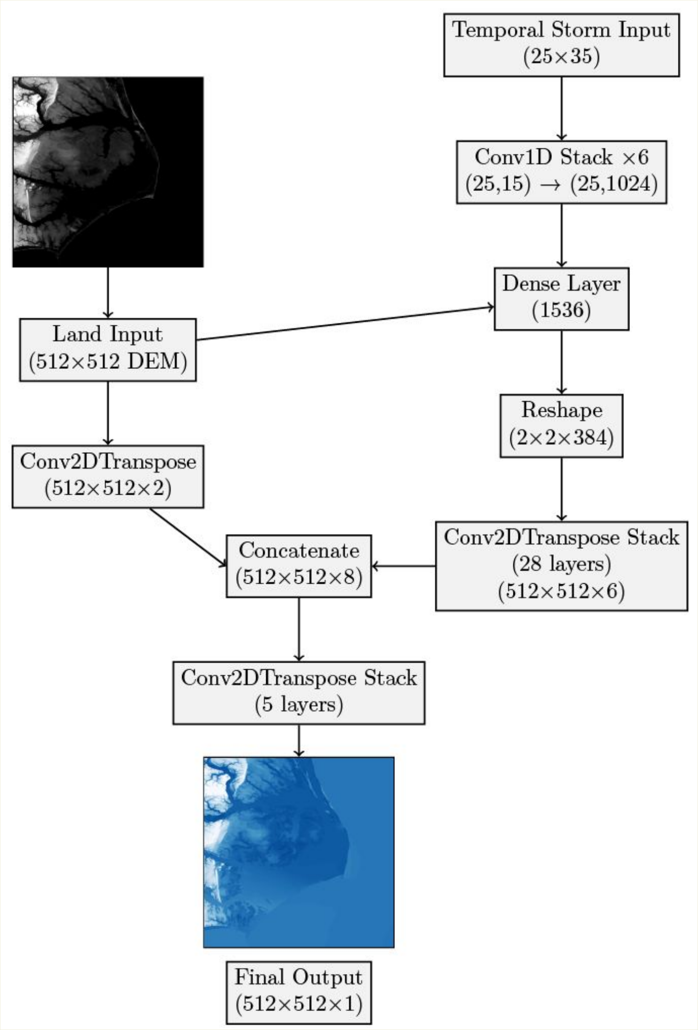
- Temporal storm-track sequence
- Initial Land elevation map for spatial context

Temporal stream:

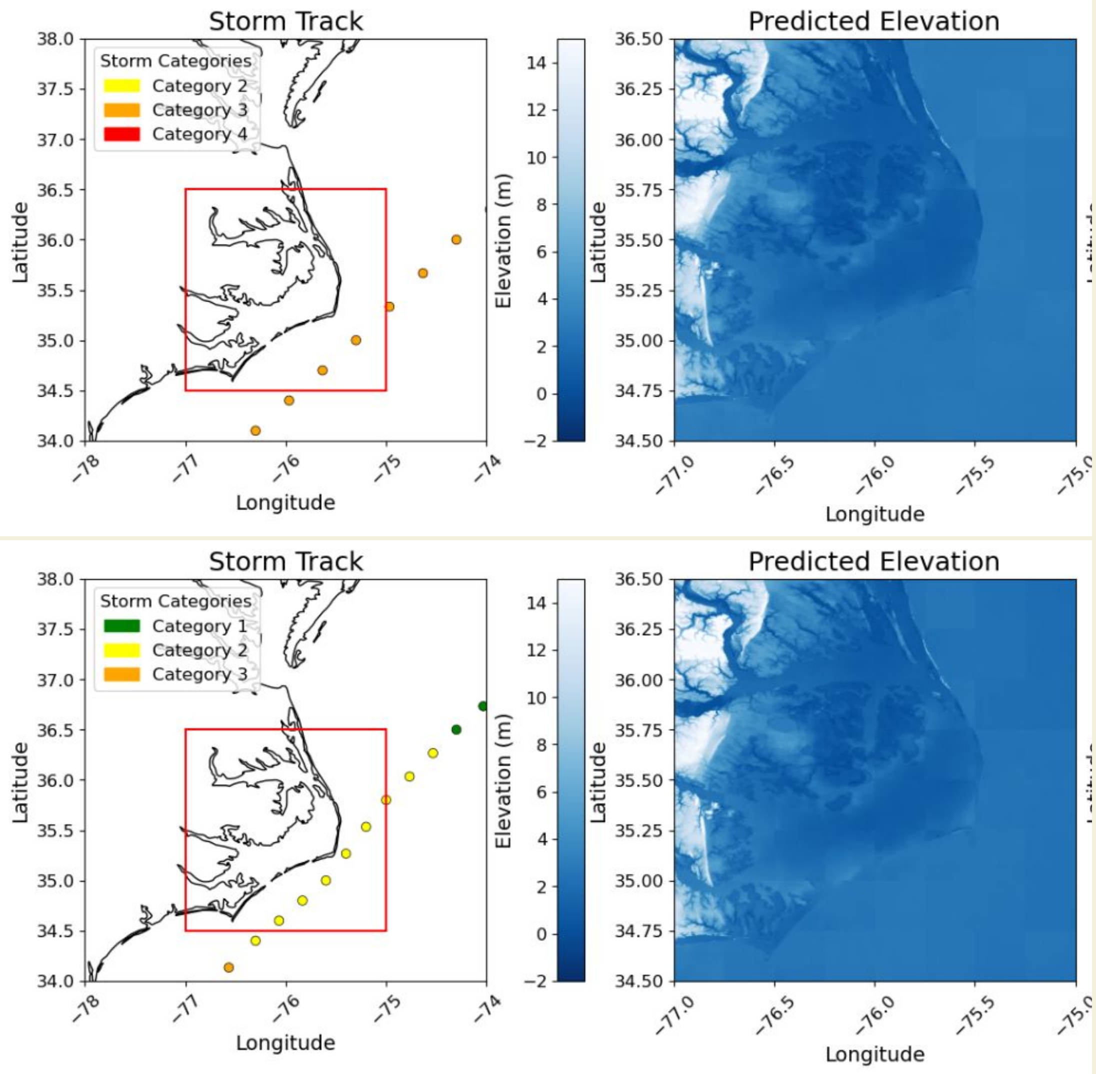
- 1D CNNs encode storm evolution and learn relationships across time
- 2x2x384 seed image output as input for spatial decoder

Spatial decoder:

- 2D ConvTranspose layers upsample from low-resolution latent → full 512×512 flood map
- High-resolution land elevation reintroduced to anchor spatial detail



Model Results



Takeaways

This model produces a spatially continuous map!

It can produce this map in 100 milliseconds

The spatial resolution of each pixel in this map (4096x4096 pixels) is about 50 m x 50 m per pixel

We are producing a high resolution storm tide map over NC in a tenth of a second!

Model Metrics

RMSE: (Root Mean Squared Error)

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

$\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n$ are predicted values
$y_1, y_2, \ldots, y_n$ are observed values
n is the number of observations

RMSE gives a higher weight to large errors.

MAE: (Mean Absolute Error)

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |\hat{y}_i - y_i|$$

Where:
$\hat{y}_i$ = Predicted value for the ith data point
y_i= Actual value for the ith data point
n = number of observations

MAE treats all errors equally regardless of magnitude.

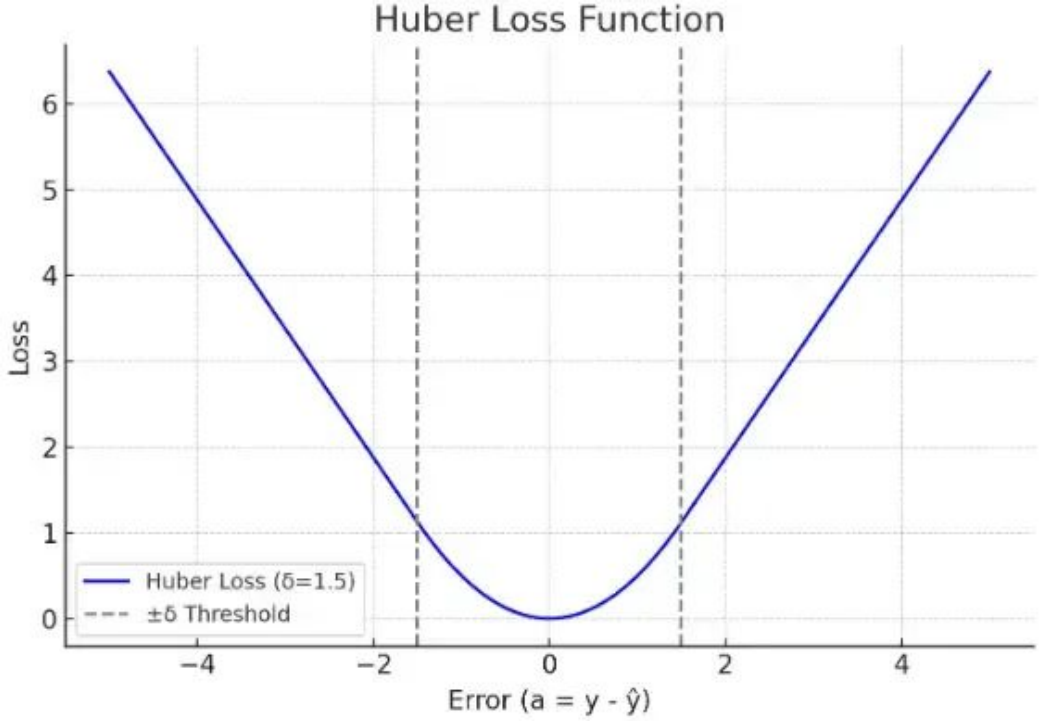
Model Loss

Huber Loss

Huber Loss = loss function combining Mean Squared Error (MSE) and Mean Absolute Error (MAE)

MSE → large elevation errors would overpower training

MAE → lose sensitivity to small elevation differences



Within ± δ the model loss function is Mean Squared Error

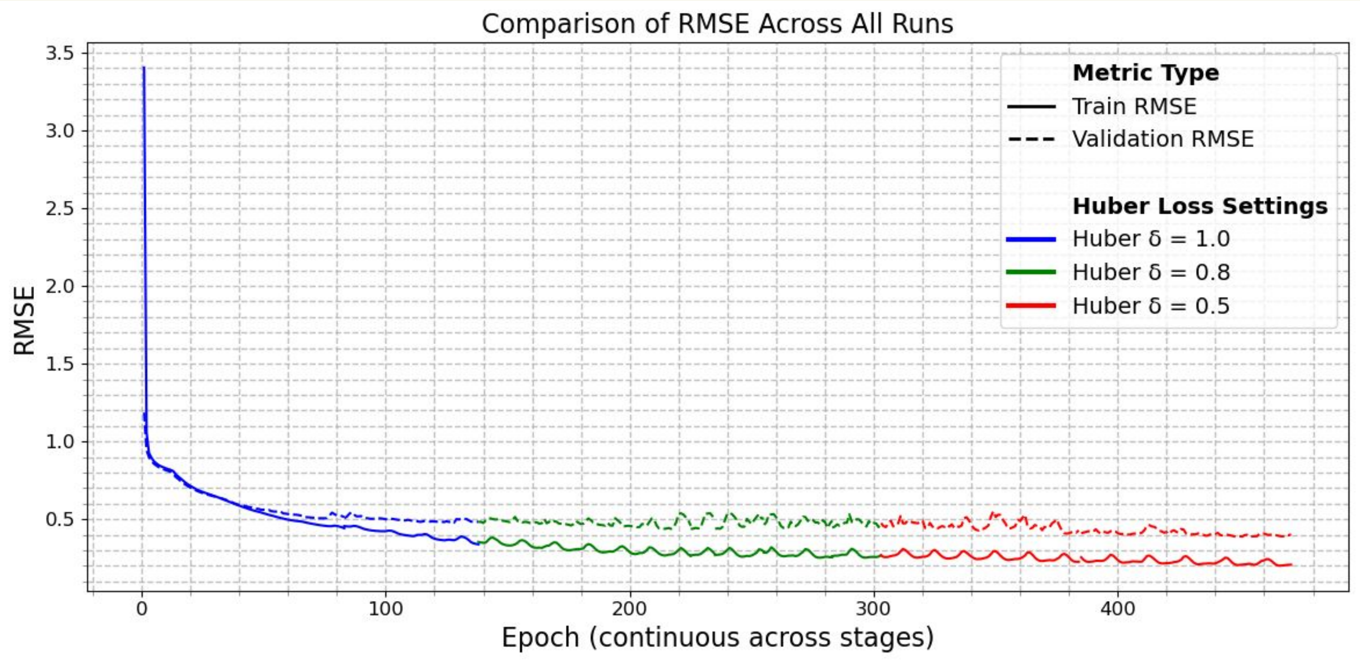
If it is larger than δ it performs like Mean Absolute Error.

Training Loss through Stages

3 stages with a change in Huber loss to help the model perform more like Mean Absolute Error

The model trained for 478 epochs : this took 370 hours

The reduction in δ makes the model more robust to outliers



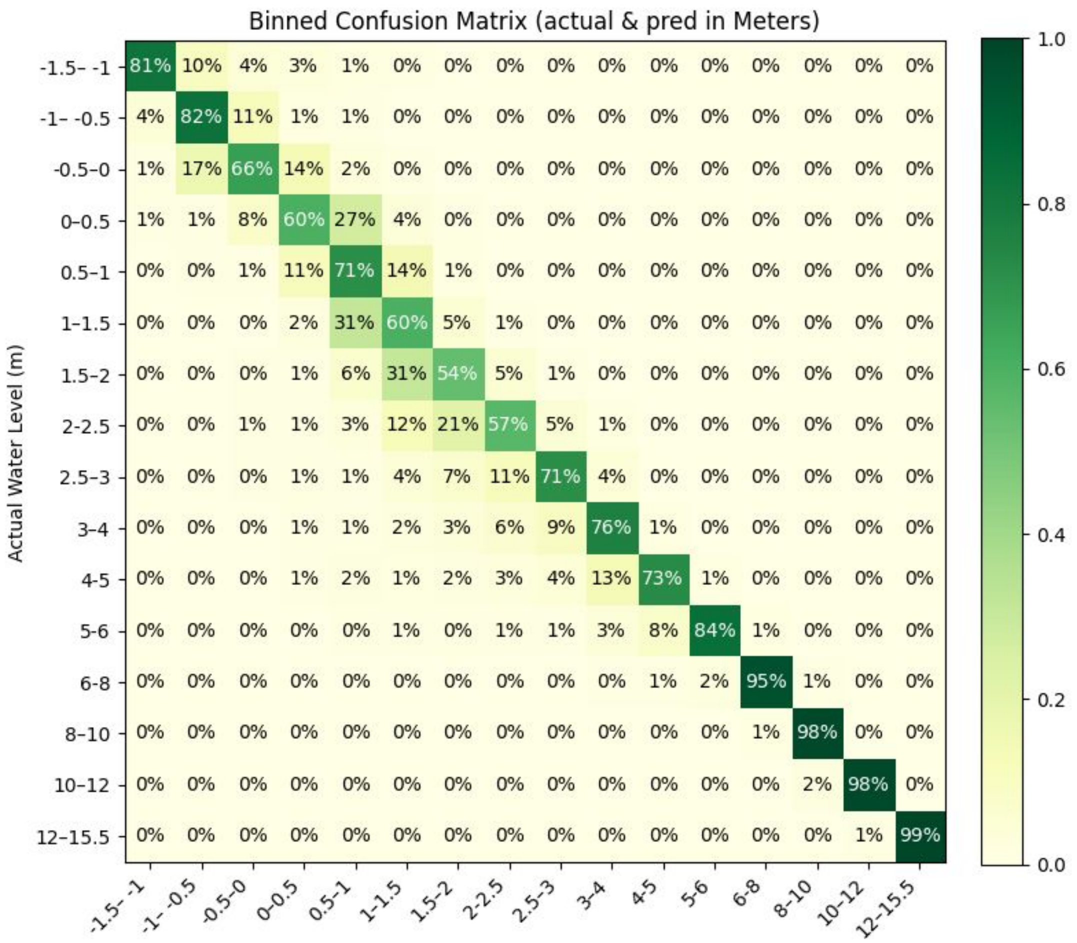
Overall Model Performance

All Test Results

Model RMSE: 0.2722 m

0–0.5 m RMSE = 0.3661 m

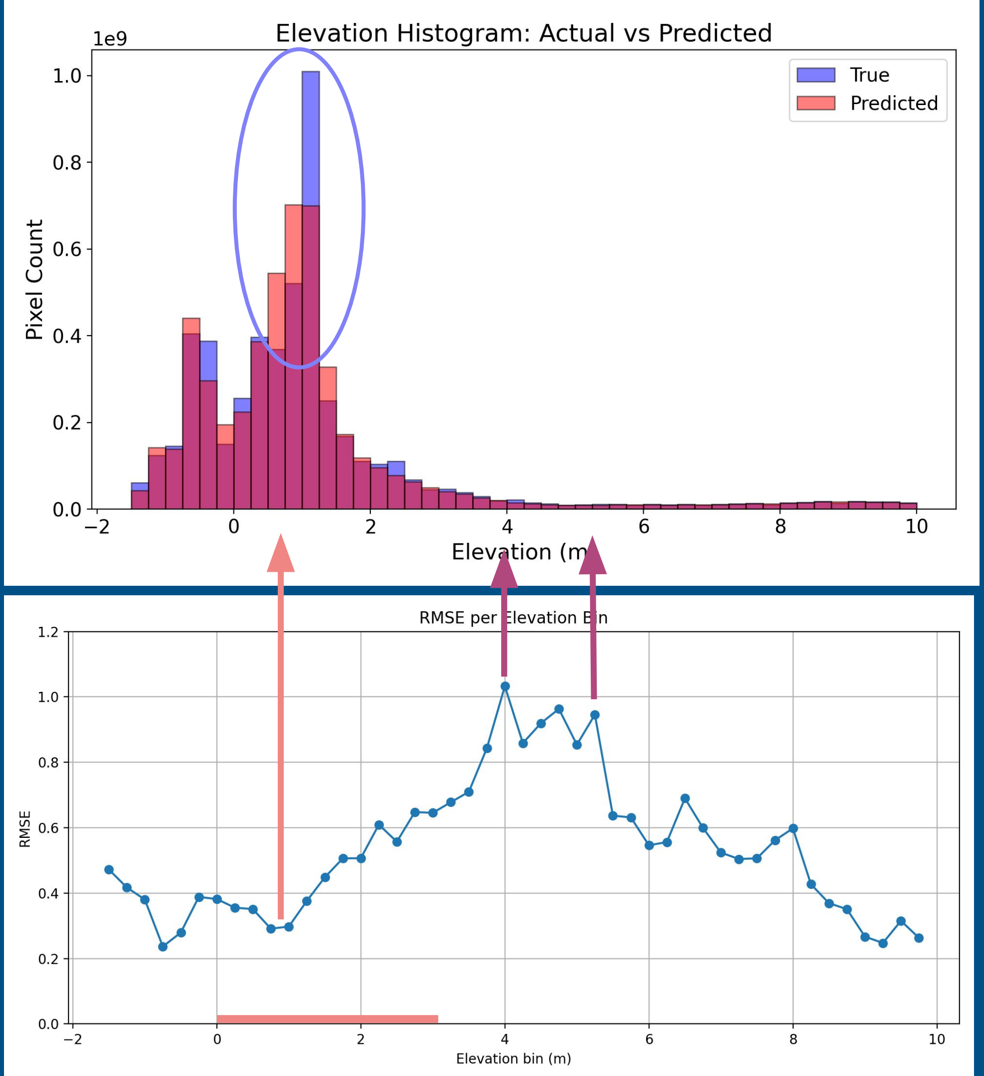
1–3 m RMSE = 0.3970 m



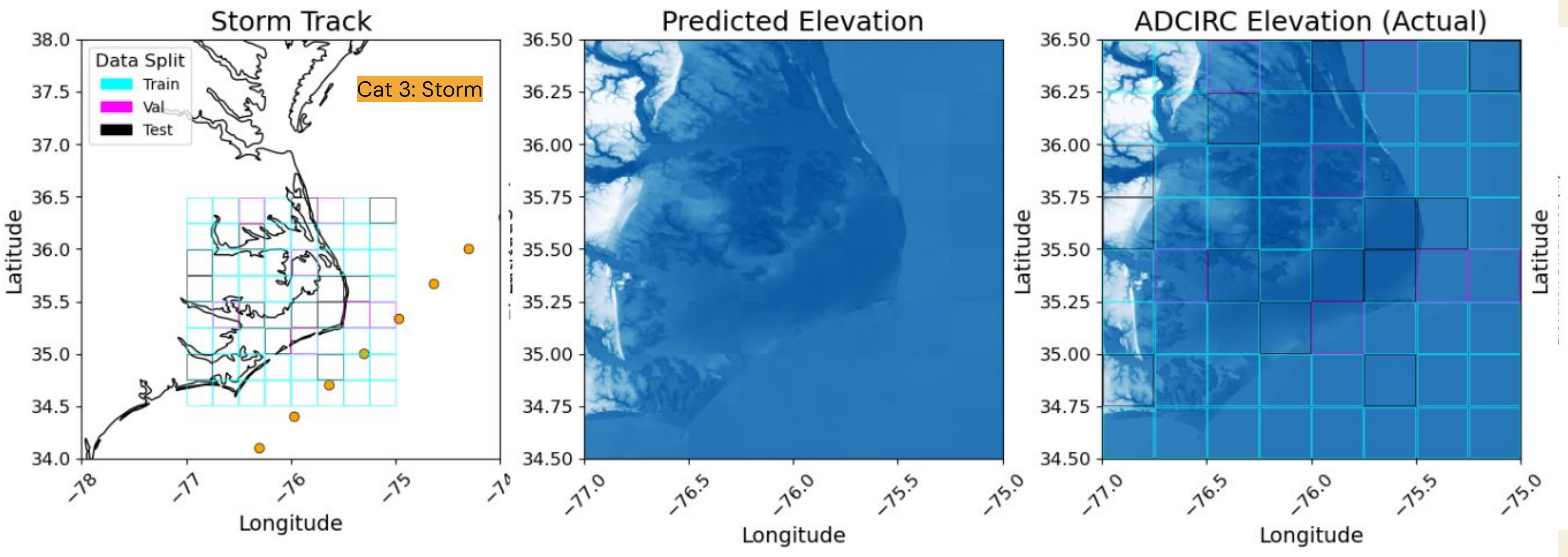
Key Takeaway: This model has high accuracy while learning the spatial structure of storm-tide flooding.

Elevation Distributions & Error Patterns

- The model preserves the large-scale elevation structure of the domain
- The histogram shows under representations in the highly dense 1 meter bin.
- Errors in RMSE peak around 1 m in the 4-5.5 m mid elevation band (low density region)
- RMSE stays low in the more densely populated regions (0-3 m) ranging from 0.4 m to 0.6 m



Domain Prediction Results



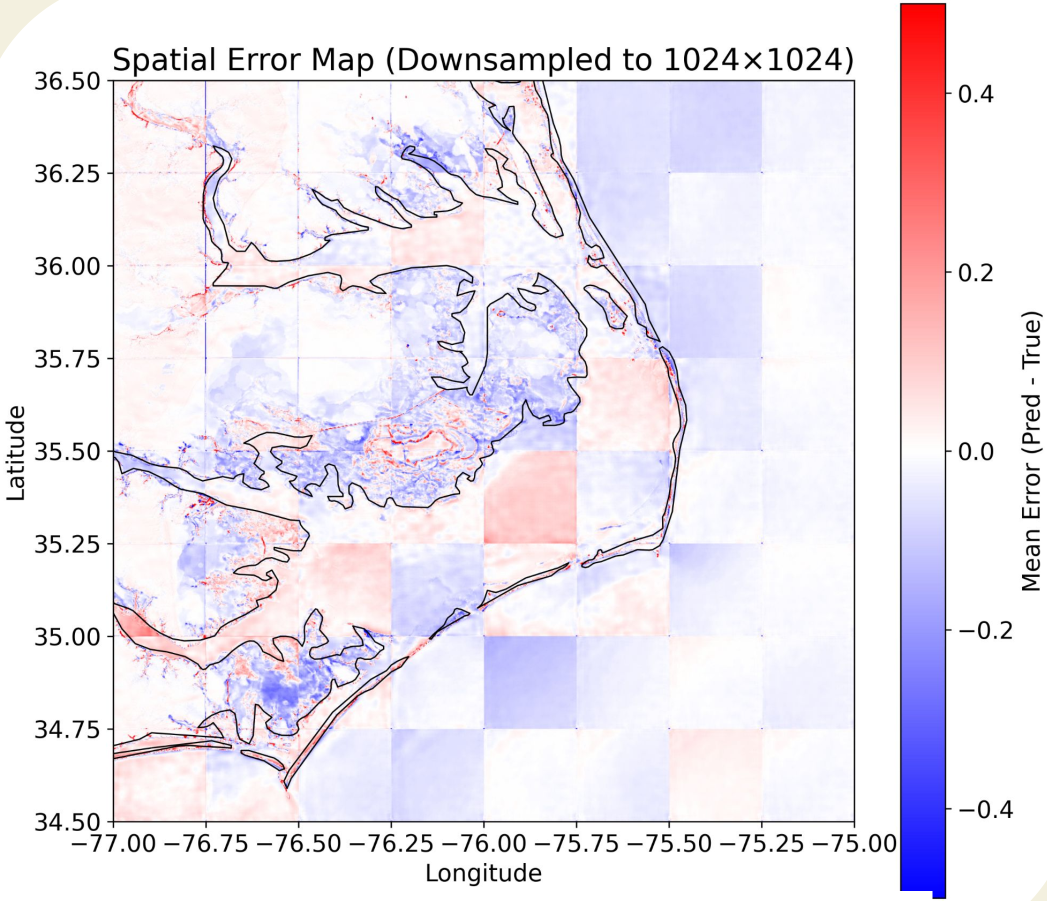
Key Takeaway: The data augmentation technique for physics aware strategy for storm-tide map tiles expands the effective training dataset while preserving the physical integrity. The model is learning how storms impact tiles.

Spatial Performance

Spatial Error Map

Most of the domain lies within 0.2 meters of the zero bias, indicating small errors and spatial consistency across the study area.

Under-estimation (–0.05 to –0.2 m) appears where steep storm-tide gradients make predictions highly sensitive to small elevation errors.



Key Takeaway: The model produces high resolution, spatially continuous storm-tide maps that describe behavior at local scales.

Comparison to Other Models

Comparison to Process-Based and AI Models

Process-Based

SLOSH (Forbes et al. 2010)

- RMSE typically 0.5-1.0 m

ADCIRC (Pringle et al. 2023)

- RMSE = 0.15–0.35 m

AI Models

Early NN surge models (Bezuglov, Hashemi)

- RMSE = 0.2–0.6 m

Hybrid Surrogates (Naeini, Pachev 2023)

- RMSE = 0.26–0.33

Storm Tide NN (Cuevas Lopez 2025)

- RMSE = 0.08–0.19m

My Model

- **RMSE=0.2722 m**

Key Takeaway: This model is highly accurate relative to process-based and AI based alternatives.

Conclusions

1. This model has high accuracy while learning the spatial structure of storm-tide flooding.
2. The data augmentation technique for physics aware strategy for storm-tide map tiles expands the effective training dataset while preserving the physical integrity.
3. The model produces high resolution, spatially continuous storm-tide maps that describe behavior at local scales.
4. This model is highly accurate relative to process-based and AI based alternatives.
5. Machine-learning surrogates can preserve physical realism when trained on high-fidelity models.
6. *Experiments with alternative model configurations provide guidance for future surrogate development. [see extra slides]*

Thank you!

Questions?

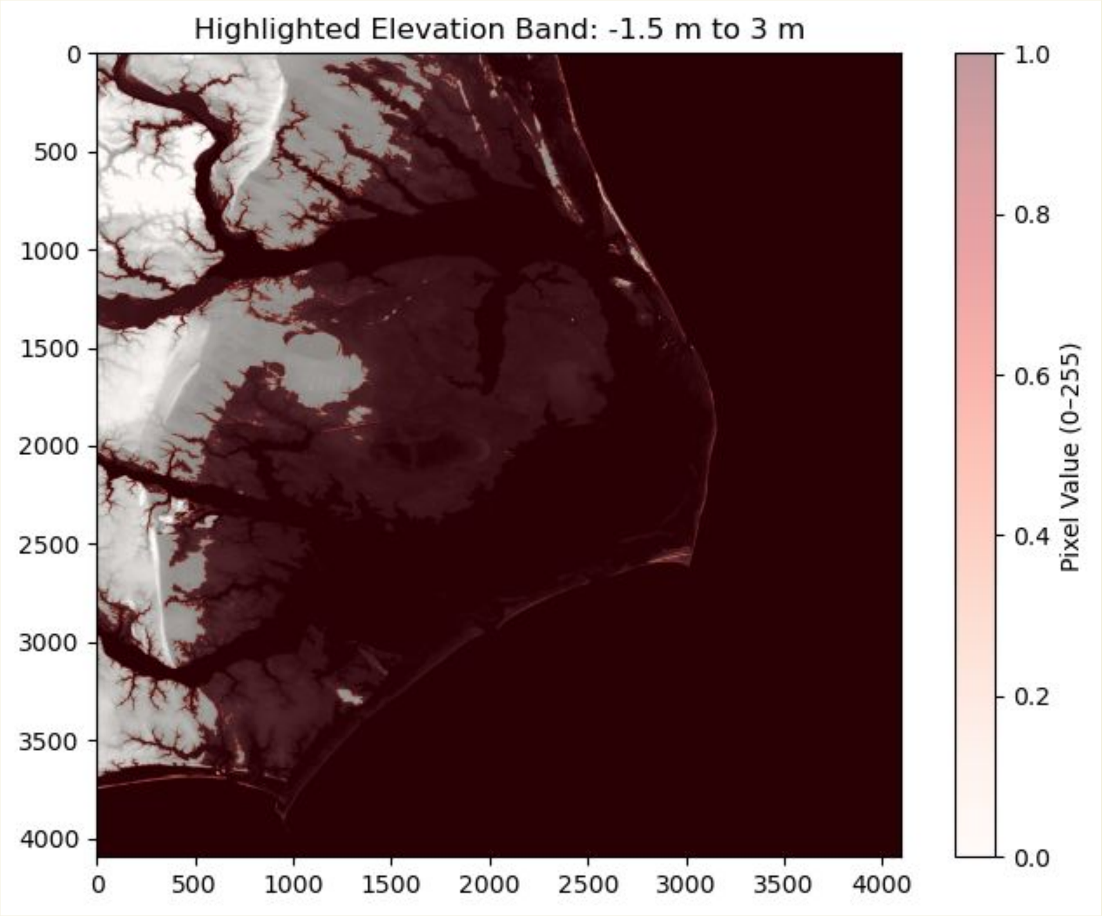
Extra Slides...

Weighted Loss Static *Model*

Weighted Loss Metric

The errors within the model are in the -1.5-3 meter range

We wanted the model to focus on this range so we implemented a weighted huber loss function for the low-lying coastal regions.

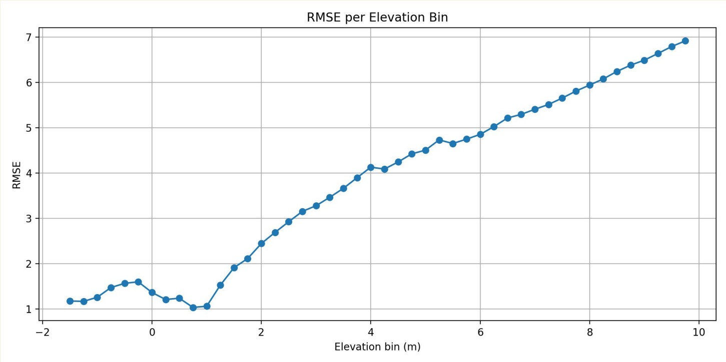


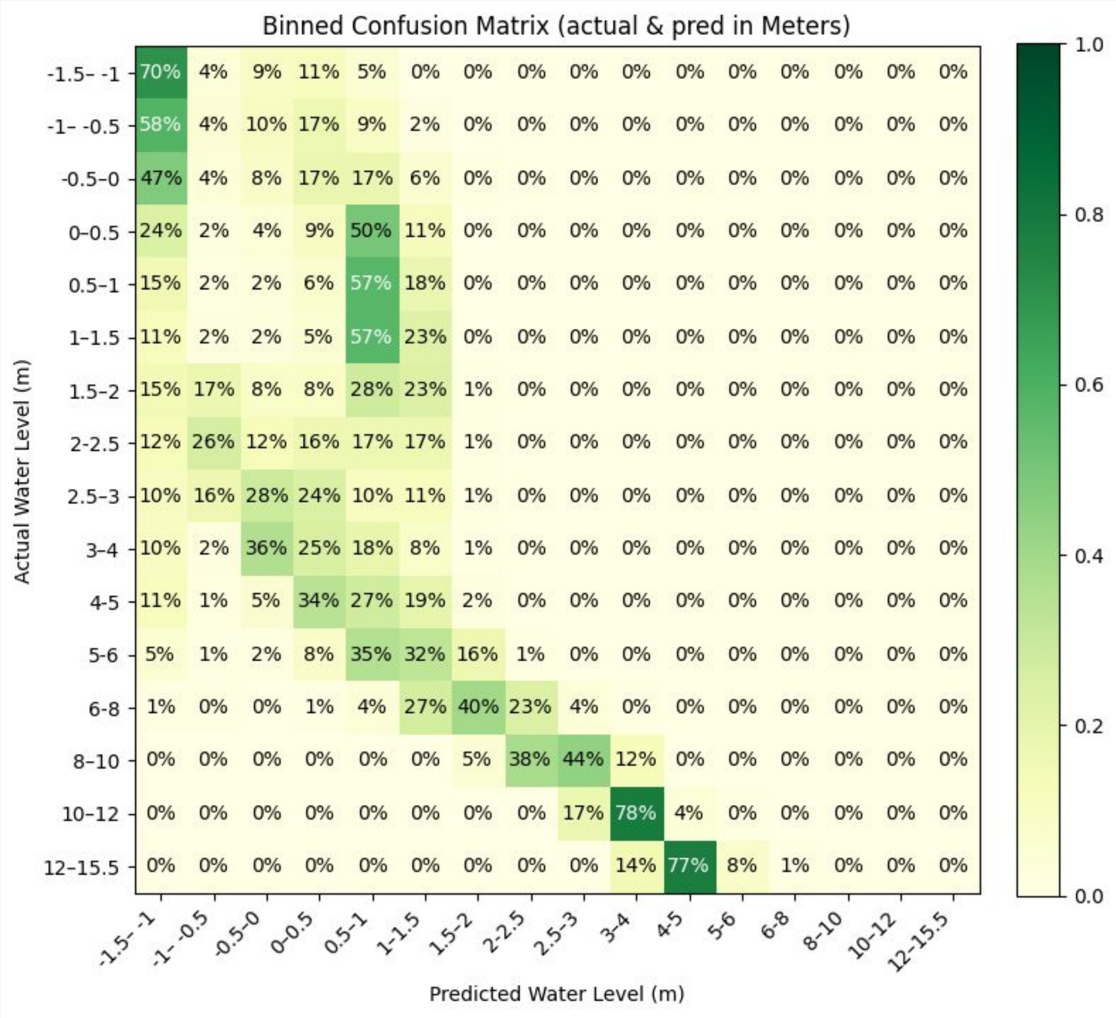
Model Performance

This model was hot started from the 139-epoch static model.

It ran for 52 more epochs but performance quickly de-stabilized.

The RMSE per elevation bin curve which stayed under 1.2 meters now peaks around 7 meters.



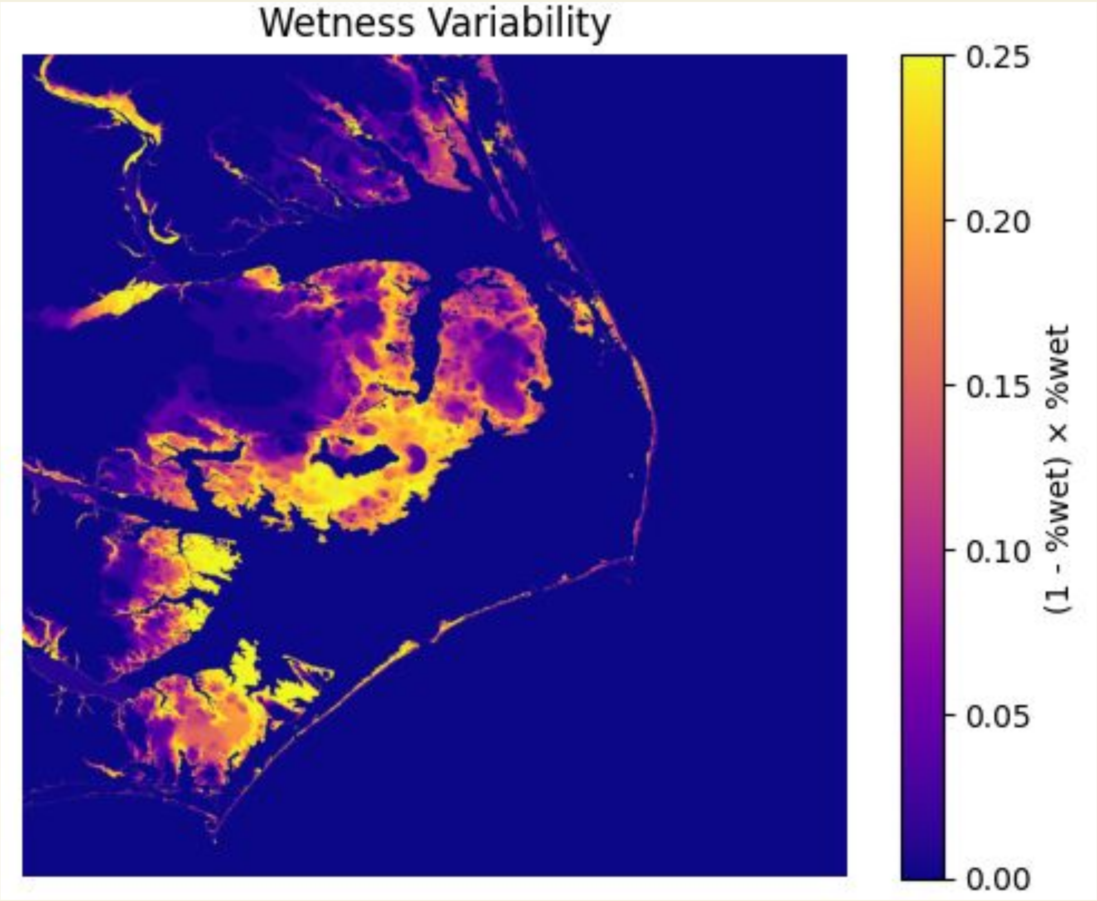


MDA Model

Wetness Variability

To determine the areas that are important for the model we computed wetness variability

The areas of high wetness variability are where the areas change the most from wet to dry



Maximum Dissimilarity Algorithm (MDA)

To select the most important a statistically significant points for the model to learn based on the wetness variability we implemented a MDA algorithm that selected image centerpoints to augment the dataset.

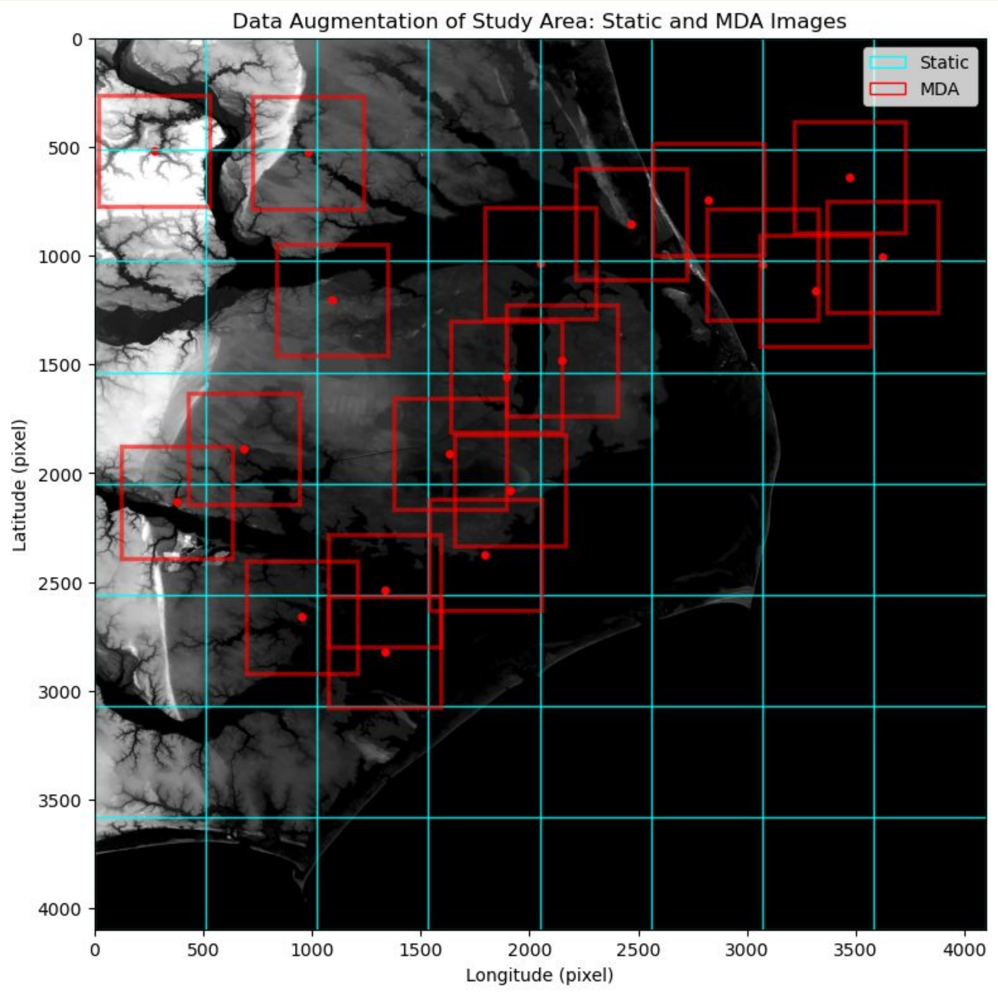
MDA Centerpoint Selection

Centerpoints for each storm were selected to be statistically relevant to the wetness variability region along with being distant from other points.

Each mda center point was checked for distance between other mda centerpoints → closest distance they could be was 250 pixels. (within one storm)

This increased the 1813 storm set into 152,292 storm maps

64 static points + 20 MDA points

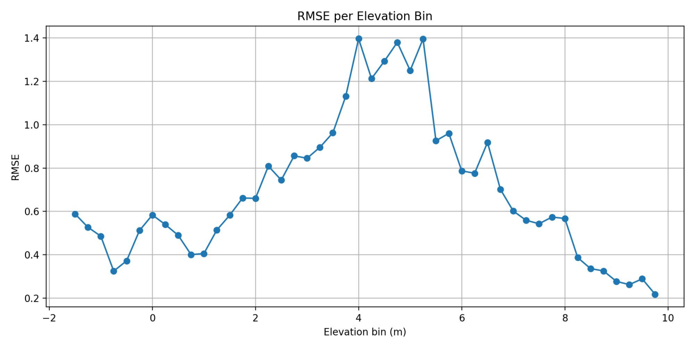


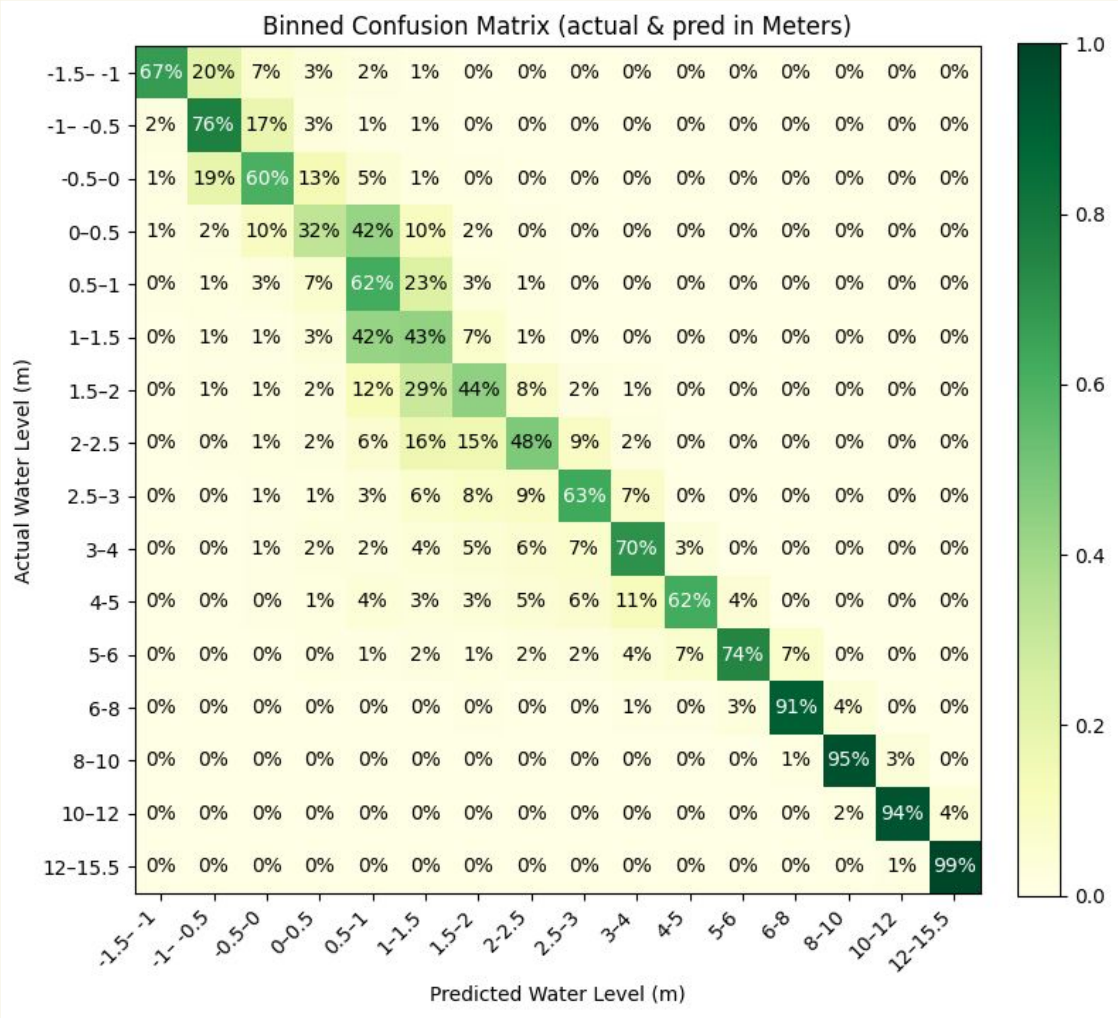
Model Performance

This model was trained for 53 epochs (4 continued runs)

This model took a long time to train almost double per time step

The overall RMSE performance of this model is around 0.40 m





MDA Shuffled Storms

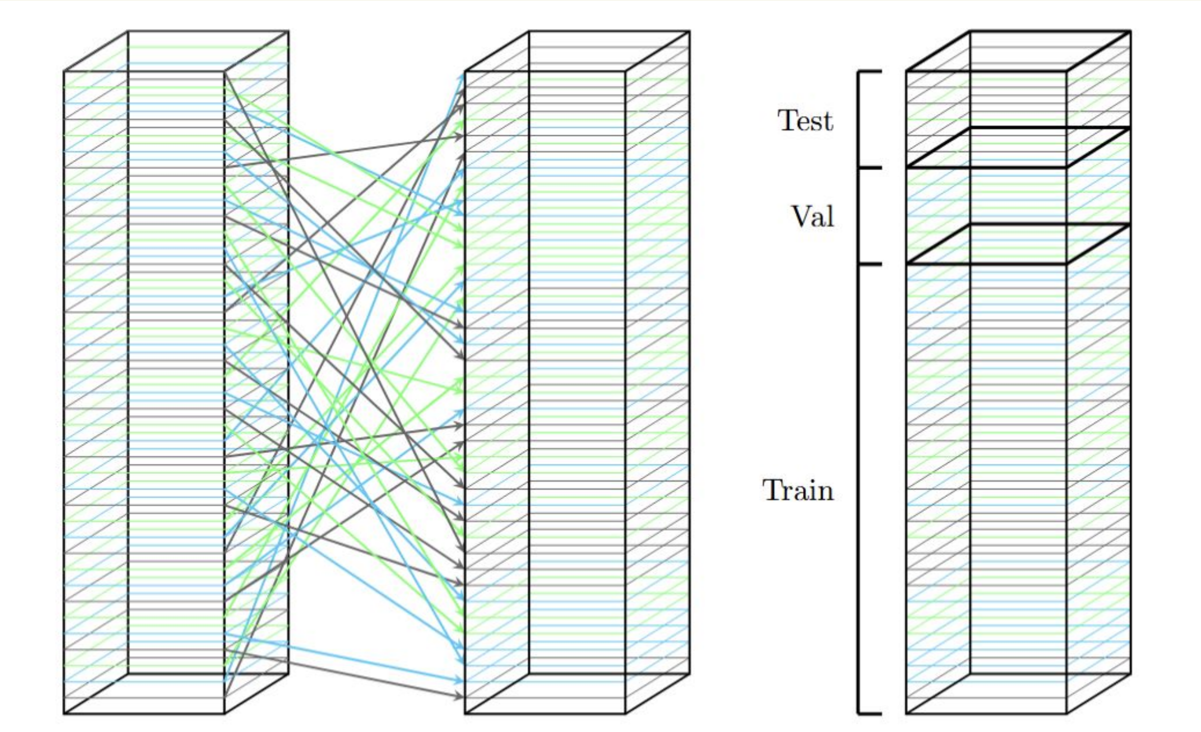
Shuffling Strategy

Two shuffling strategies:

- Fully randomized shuffling
- Shuffling storms and then shuffling within test, val, train

Storm shuffling gives the model the ability to see the entire storm in training or validation dataset.

This would also check the models ability to generalize on storms it has not seen before in the Test dataset



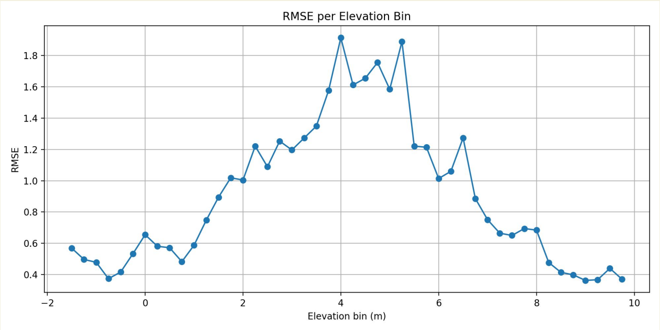
Model Performance

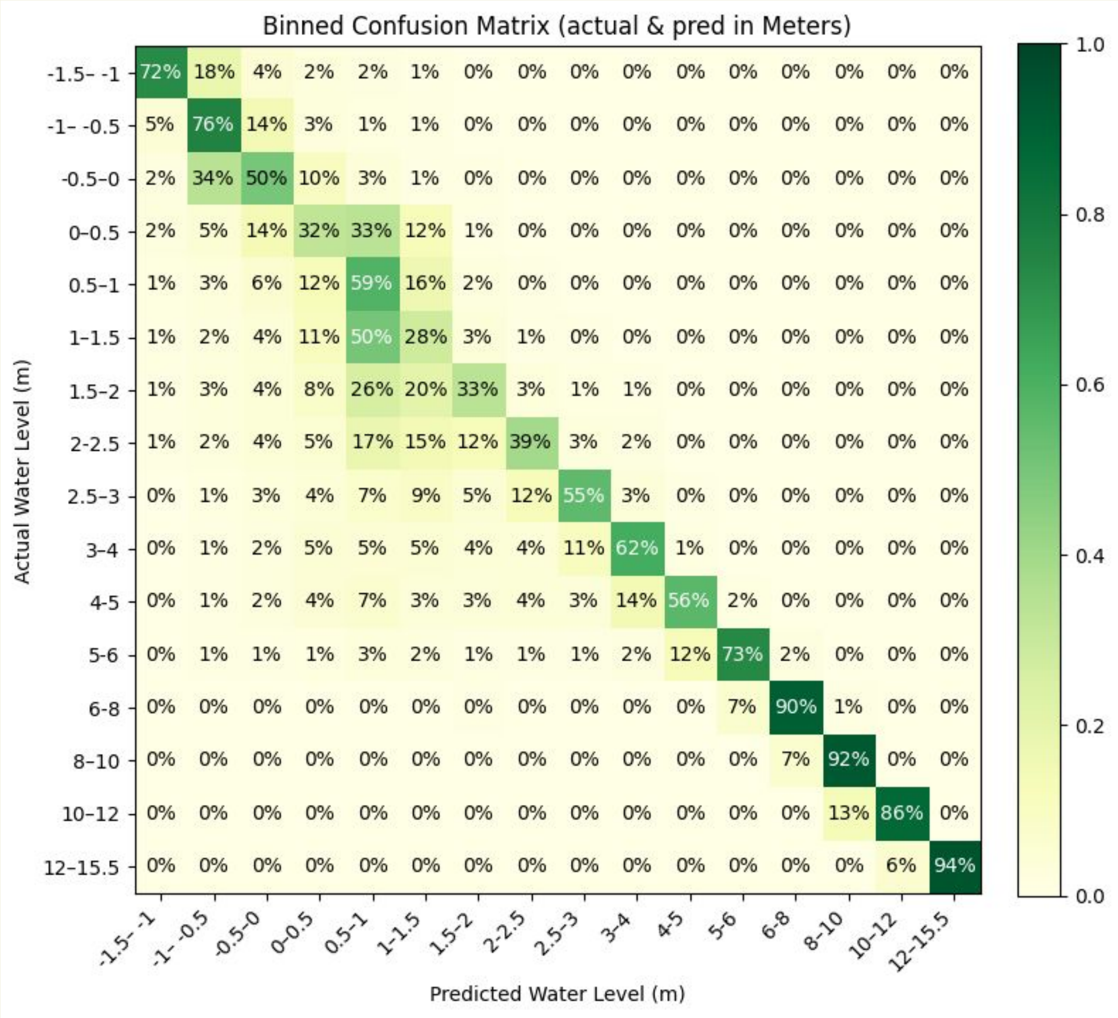
This mda shuffled storm model trained for 171 total epochs

RMSE performance was around 0.45 m

Similar to the other MDA model it saw higher RMSE per elevation bins.

Additionally the confusion matrix shows under predictions in the 0–2 m bins





Extraneous Method Slides

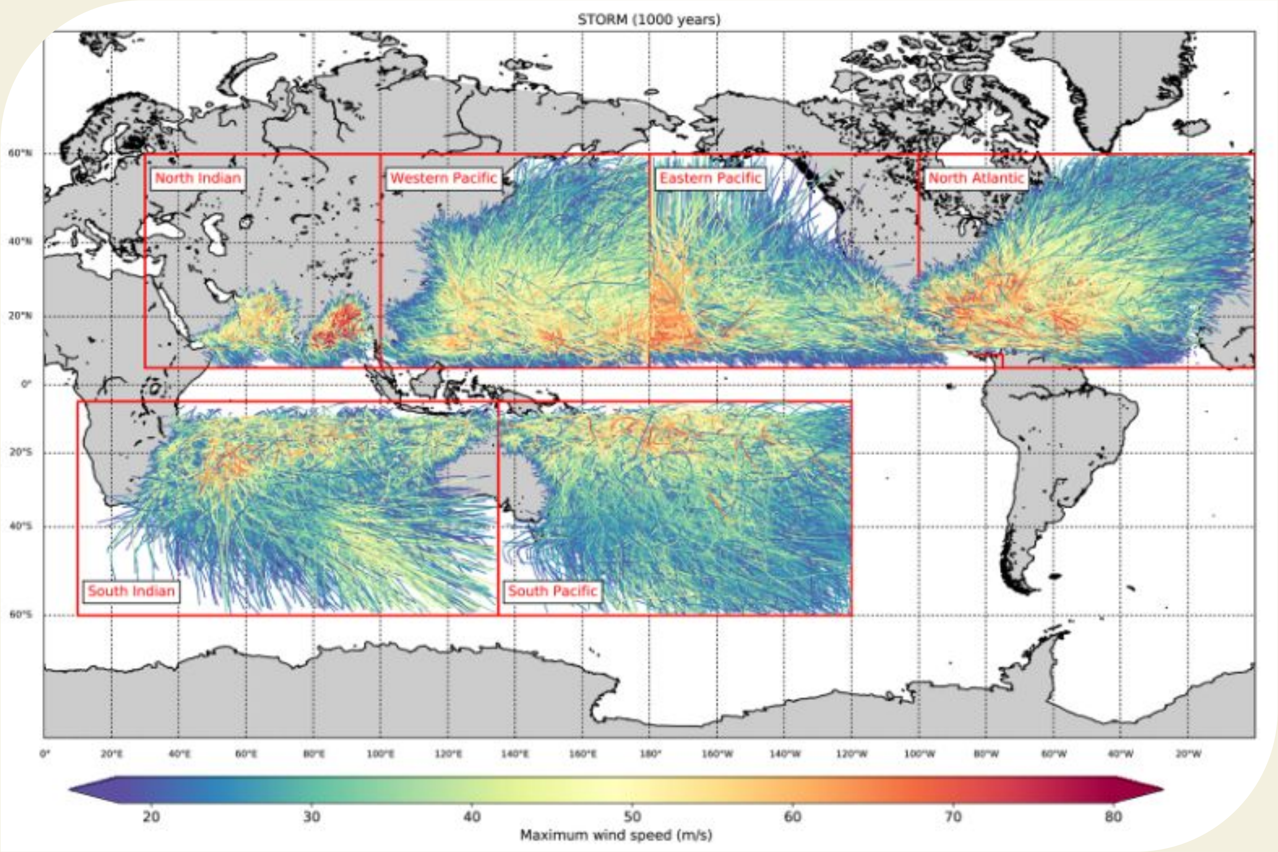
S.T.O.R.M. Library

Statistical extension of the IBTrACS historical dataset

Generates ~100,000 synthetic storms within realistic physical bounds

Preserves historical storm characteristics and variability

Separated by basin to maintain regional climatology



The STORM library separated into storm basin. [Bloemendaal et al.]

Dense Layer & Reshape

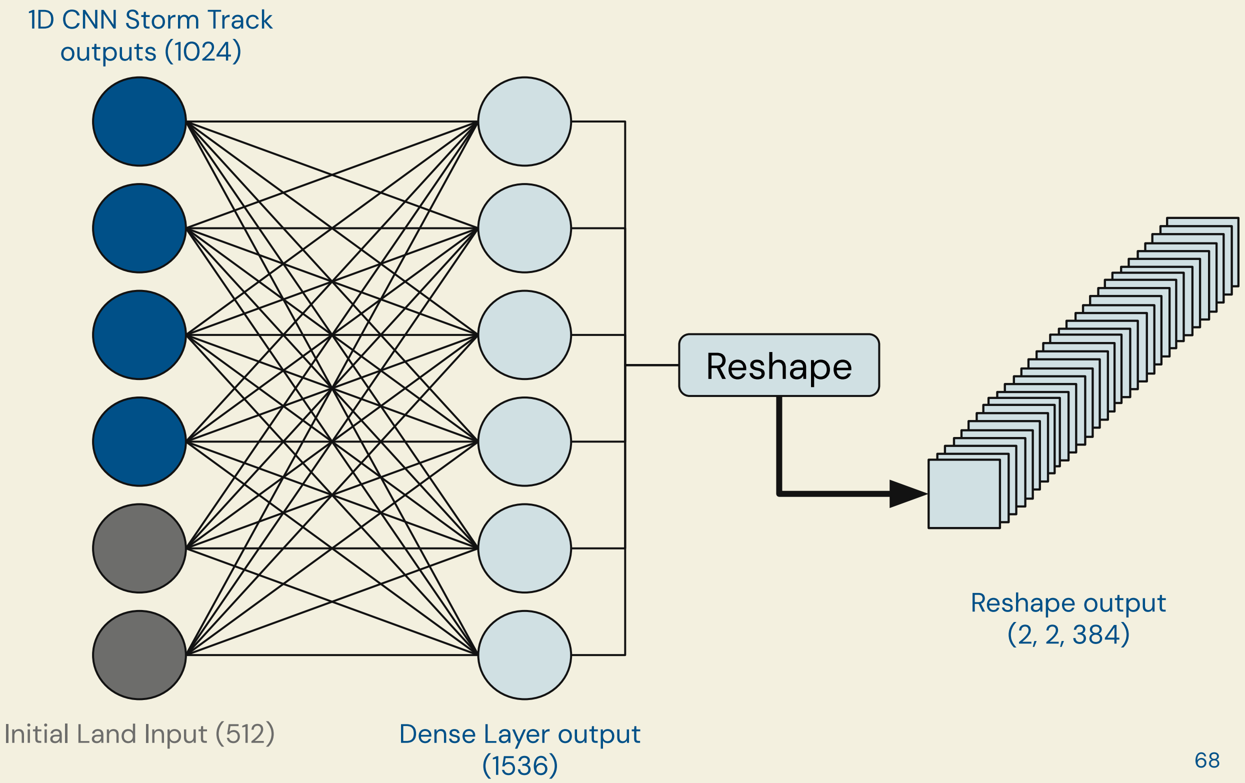
1D-CNN storm-track encoder outputs a 1024-dimensional feature vector

Initial land elevation features (512 values) are concatenated with the storm features

Dense layer combines these into a fused 1536-dimensional representation

Dense output is reshaped into a small spatial tensor: (2, 2, 384)

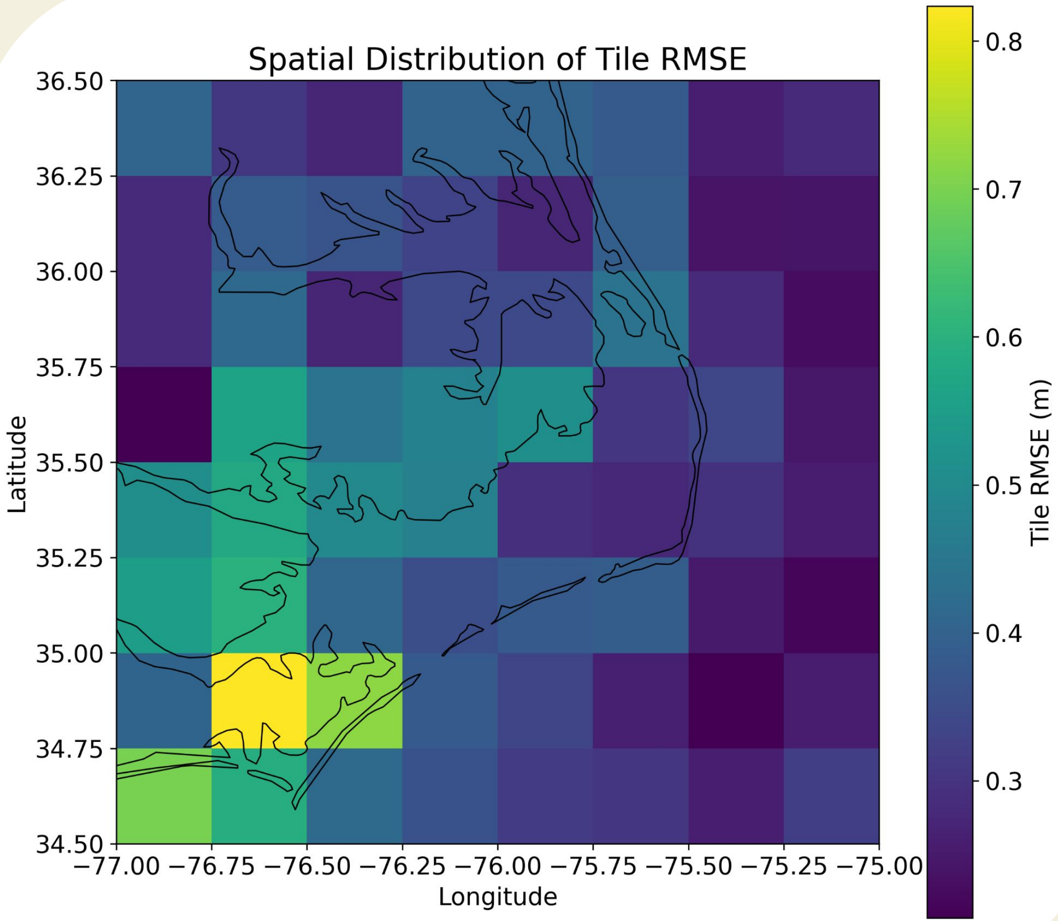
This 2×2×384 tensor becomes the starting point for 2D CNN upscaling

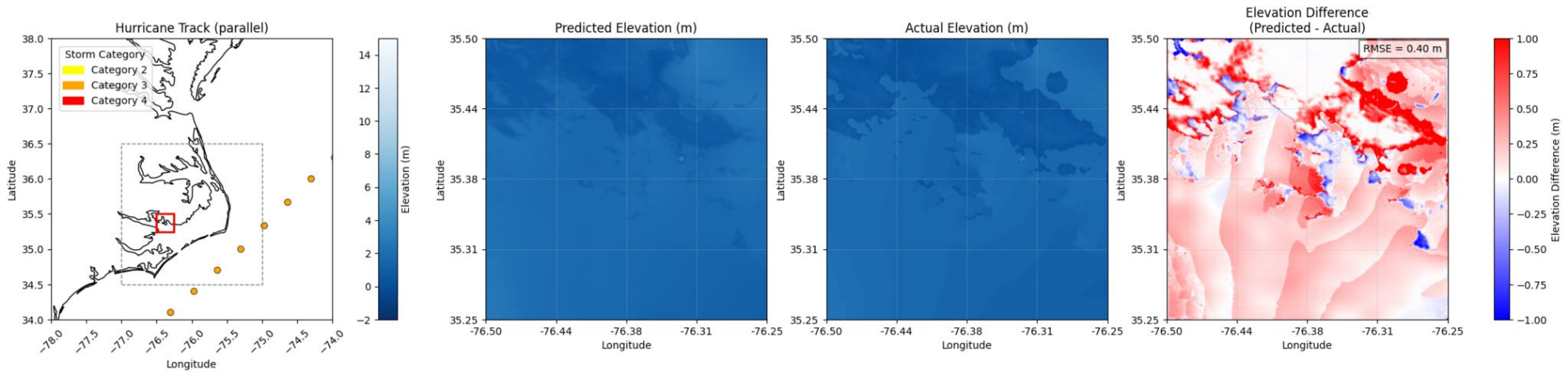


Spatial RMSE

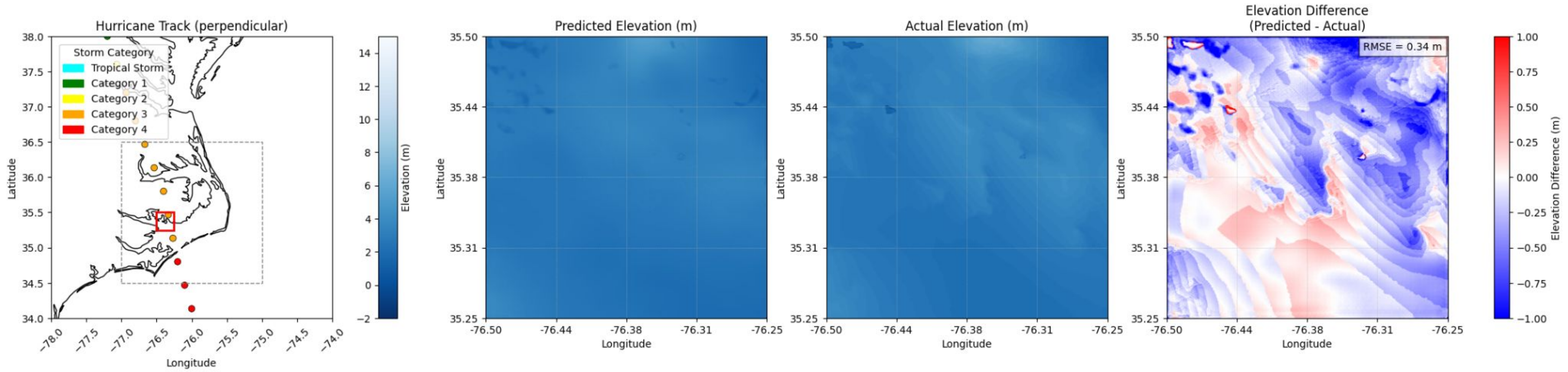
RMSE stays low across the domain (0.30–0.45 m)

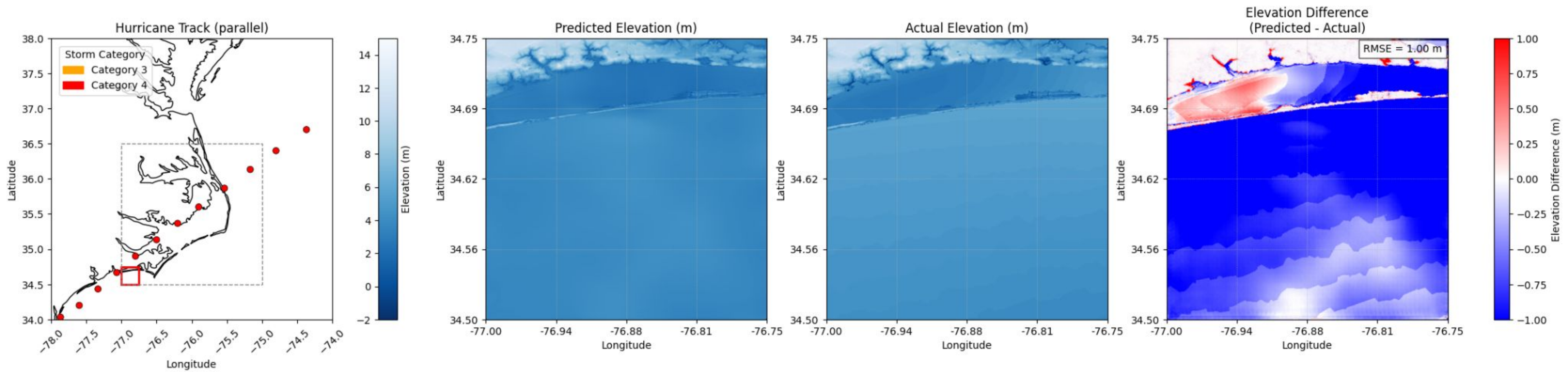
Elevated errors (0.7–0.8 m) in nonlinear, topographically complex regions



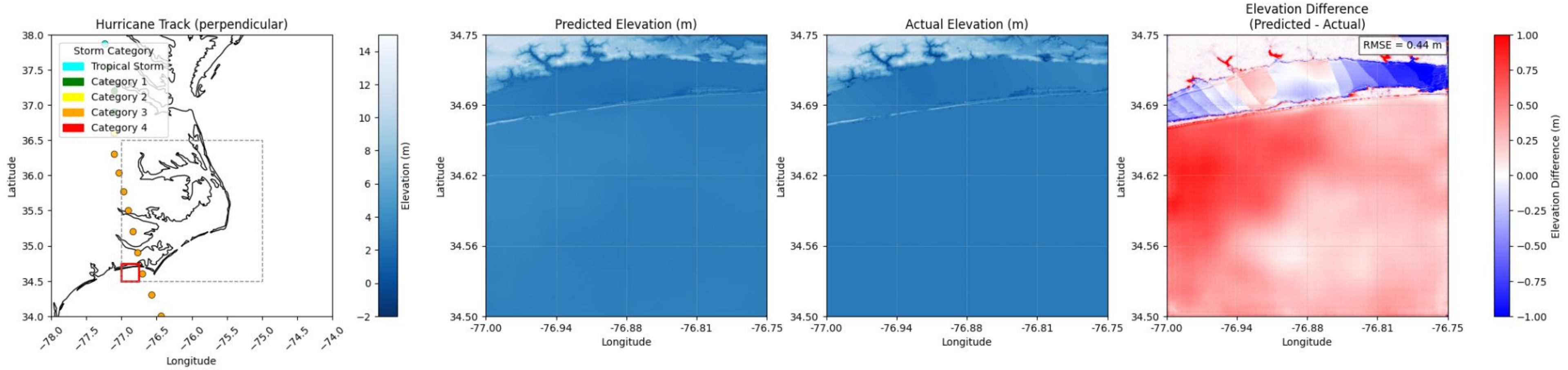


Perpendicular vs Parallel Storms





Perpendicular vs Parallel Storms



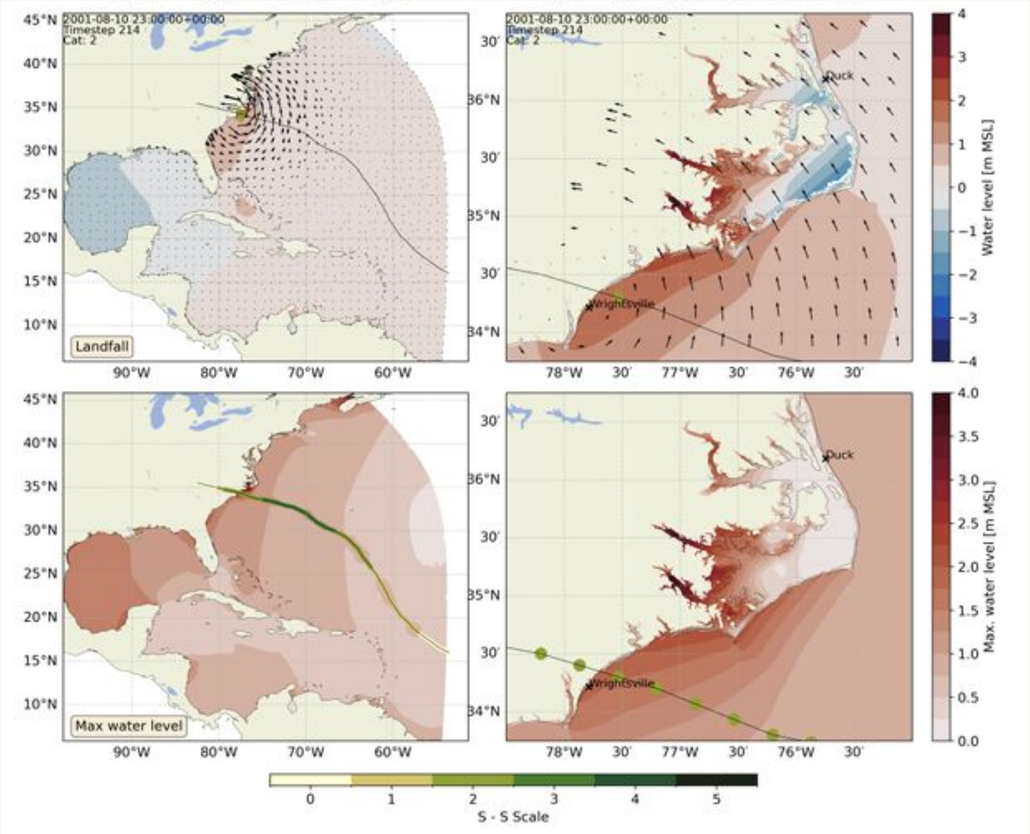
‘**Elevation**’ refers to either the storm-tide surface elevation (in wet regions) or the ground-surface elevation (in dry regions), whichever is higher.

ADCIRC Outputs

ADCIRC produces total water levels on an unstructured mesh

We extract the maximum water elevation for each storm
- Peak storm-tide response at every mesh node
- One mesh-based field per storm

High-resolution values at all ADCIRC mesh vertices



ADCIRC Storm 0 impact on full domain and NC [Cuevas Lopez et al. 2025]

How can we quickly and accurately predict storm flood maps?

SLOSH

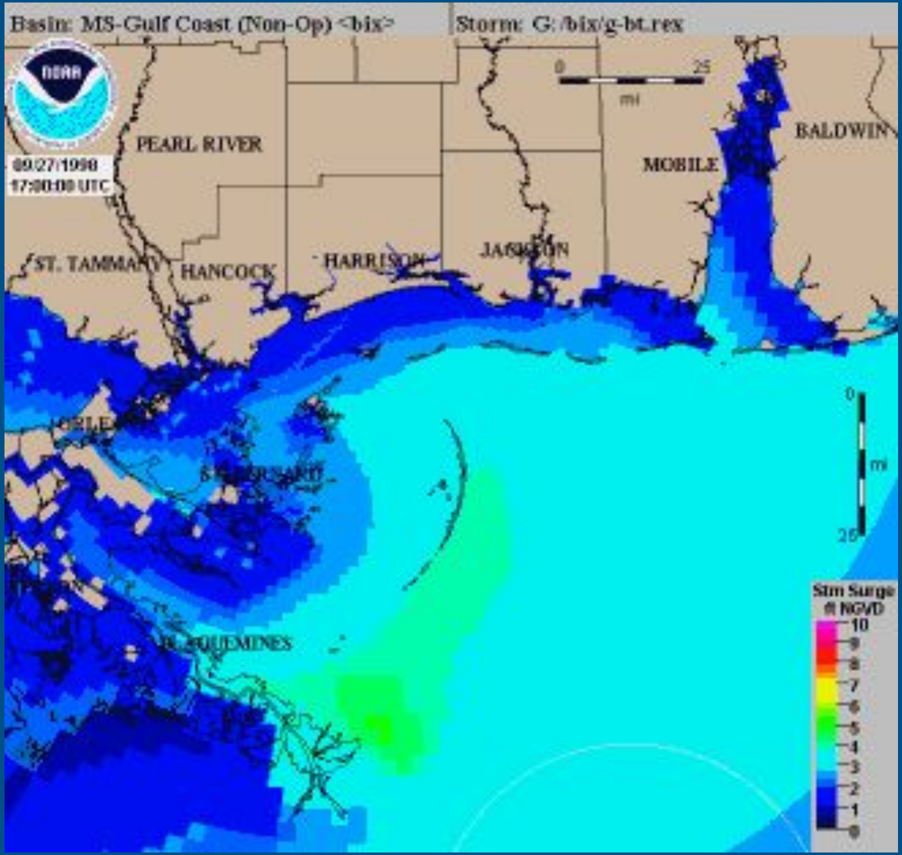
Solves a simplified form of the shallow-water equations

Neglects nonlinear advection; uses 1-D derivations

Coarse wind and pressure fields on a basin-scale grid

Very fast, but lower spatial accuracy (Kerr et al., 2013)

Uses large ensembles to compensate for uncertainty



Development of a hurricane by SLOSH model run [NOAA]

Accuracy vs Speed

ADCIRC is highly accurate, and can produce high resolution results.

ADCIRC speed depends on:

- Mesh Resolution
- Number of Cores
- Length of simulation

High computational cost

SLOSH is very fast, running in minutes to hours.

SLOSH inaccuracy is due to:

- Simplified equations
- Simplified wind pressure forcing
- Coarse basins

SLOSH can be run probabilistically but it still does not reach the accuracy of ADCIRC

If accuracy and speed are both essential, how do we build a model that delivers both?

Neural Network Architecture