Dear Editor and Reviewer,

Thank you for the constructive feedback regarding our submission. We have revised the manuscript based on the suggestions. Detailed point-by-point response to the reviewer's comments follows below.

COMMENT 1:

Review of "HIDRA-D: deep-learning model for dense sea level forecasting using sparse altimetry and tide gauge data"

OVERVIEW

This paper presents an improvement to the existing HIDRA sea level forecasting model. The paper is interesting and rigorous, and the writing is clear. Aside from a few points and methodological steps that require further elaboration, the manuscript is well suited for publication. I suggest a minor revision.

RESPONSE 1:

We sincerely thank the reviewer for the positive assessment of our work and the encouraging feedback. We have addressed the specific methodological questions and the suggestions regarding the discussion of high sea level performance.

COMMENT 2:

HIGH SSH PERFORMANCE

From Table 3, it is apparent that HIDRA-DN performs markedly worse than NEMO for high SSH levels, which are usually the ones we care about most from an impact perspective. More discussion of this point is warranted than the existing note on line 354. This suggests that HIDRA-D is not "better" than NEMO at least for some applications, particularly coastal flood warning, a point which should be highlighted in the conclusions. Do you have an understanding or theory as to why the model performs worse on high SSH? Might there be a way to correct it that would not harm the model's overall skill?

RESPONSE 2:

We agree that distinguishing between overall basin performance and the specific capability to predict coastal extremes is important, particularly for applications like flood warning. We have revised the manuscript to explicitly address this limitation in both the Discussion and Conclusions sections. Specifically, we expanded the discussion to detail why HIDRA-D^N underperforms in these scenarios, listing three factors: the spatiotemporal sparsity of satellite ADT data which rarely captures transient storm surges; the regression-to-the-mean tendency of MSE-based training in data-sparse regimes; and the lack of explicit in-situ undisturbed water depths to resolve local topographic effects on sea level at untrained locations.

Regarding the possibility of correcting this behavior, we found that standard approaches like weighted loss functions (prioritizing extreme values) did not yield improvements in our experiments. We suspect this limitation is inherent to the current approach, specifically the reliance on sparse satellite altimetry which lacks sufficient dense ground truth for coastal extremes. We updated the conclusions to clarify that while HIDRA-D excels at basin-scale dynamics, traditional numerical models remain superior for coastal flood warnings at locations where no prior training tide-gauge data exists.

Changes in *Performance along the coastal region*:

In contrast, for high sea level values, NEMO outperforms HIDRA-D^N. Figure 10 shows RMSE scores for each tide gauge location for further insights. While HIDRA-D^N achieves comparable error levels to NEMO at many locations, it exhibits significantly higher RMSE at Koper, Venice, and Neretva for high sea level values. This discrepancy likely arises from three connected reasons. First, the primary supervision for the dense field comes from satellite ADT data, which is spatiotemporally sparse (Fig. 3); the probability of a satellite track capturing the peak of a transient short-duration storm surge is low, leading to a training set imbalanced against extreme dense events. Second, deep learning models trained with mean squared error objectives tend to produce smoothed outputs to minimize global error, occasionally underestimating sharp peaks (regressing to the mean). Third, extreme values at specific locations are often driven by unresolved local topographic effects in bays and harbors. In the leave-one-out HIDRA-D^N setup, the model must infer these local dynamics without ever seeing training data from that specific coastline geometry, whereas NEMO explicitly solves physical equations using high-resolution bathymetric grids. These findings suggest that HIDRA-D^N is highly effective for open waters and general basin dynamics but faces limitations in resolving localized coastal extremes at locations where it has not been explicitly trained.

Changes in *Conclusions*:

computationally expensive numerical models for sea level forecasting. Furthermore, the model exhibits remarkable robustness to a sparse tide gauge network, successfully capturing large-scale dynamics even when trained on remote stations, which is critical for applications in data-sparse regions. However, although HIDRA-D, like NEMO, captures large-scale sea level trends, it struggles to reproduce high-frequency local variations and extreme peaks at untrained coastal locations. The model's performance is highest in open waters but degrades in coastal areas with complex bathymetry, such as Koper and Venice, specifically during extreme sea level events. Consequently, while HIDRA-D offers a computationally efficient alternative for basin-scale forecasting, it currently lags behind traditional numerical models like NEMO for specific applications such as coastal flood warnings at locations where no prior training data is available.

COMMENT 3:

MINOR POINTS

Line 218: “The resulting field is then adjusted using the land-sea mask for the Adriatic Sea”. Can you be more specific what this adjustment is?

RESPONSE 3:

The adjustment is element-wise multiplication of the predicted 2D field with a static binary mask. Since the IDFT generates values for the entire rectangular grid, this masking step is necessary to set all values corresponding to land points to zero. Our specific mask is included in the published dataset.

The text now reads “The resulting field is then multiplied by a binary land-sea mask of the Adriatic.”

COMMENT 4:

Line 275: Describe in more detail how you arrived at these weights. Was there an objective hyperparameter tuning? Or hand-tuned? If so, what metric(s) were you trying to maximize during this tuning?

RESPONSE 4:

Note that the weights were not determined through a numerical hyperparameter optimization process. Instead, they were chosen based on a specific architectural design choice to enforce a hierarchical prioritization of tasks. Our primary objective was to ensure that the HIDRA3 backbone, which generates the latent representations, retains its high accuracy for point-wise tide gauge predictions (supervised by L_1). If the weights were of similar magnitude, there would be a risk that the optimization process might degrade the accuracy of the HIDRA3 backbone (point predictions) in an attempt to minimize the dense reconstruction error (L_2 and L_3). By setting $\alpha = 100$, we impose a soft constraint that forces the backbone to prioritize the point-prediction task. This ensures that the dense predictions are constructed on top of a valid underlying state representation, rather than compromising that state to fit the sparse satellite data.

Regarding $\beta = 1$ and $\gamma = 1$: L_2 and L_3 generally operate on spatially distinct domains. L_3 provides supervision at fixed coastal points where satellite data is rarely available, while L_2 provides supervision in open water. Since they do not compete for the same spatial grid points in most iterations, equal weighting was sufficient.

We have revised the section in the manuscript to explain this design logic:

$$\mathcal{L} = \alpha\mathcal{L}_1 + \beta\mathcal{L}_2 + \gamma\mathcal{L}_3. \tag{7}$$

The weights were selected to enforce a hierarchical training structure: $\alpha = 100$, $\beta = 1$, and $\gamma = 1$. The significantly higher magnitude of α is a design choice intended to act as a soft constraint, ensuring that the HIDRA3 backbone prioritizes the accuracy of point-based SSH predictions ($\mathcal{L}_1$) above all else. This prevents the optimization of the dense reconstruction losses ($\mathcal{L}_2$ and $\mathcal{L}_3$) from degrading the quality of the underlying backbone representations. Consequently, the backbone learns primarily from the high-fidelity tide gauge data, while the Dense decoder adapts to these representations to satisfy the basin-wide constraints. The weights β and γ are set equally as $\mathcal{L}_2$ and $\mathcal{L}_3$ operate on largely spatially disjoint sets of points (sparse satellite tracks versus fixed tide gauge locations) and therefore do not require competitive weighting.

COMMENT 5:

Line 281: "To progressively decrease the learning rate by a factor of 100, we utilize a cosine annealing schedule". Unclear. Does this mean that the starting and ending LR during training differ by a factor of 100? Or something else?

RESPONSE 5:

As suggested, we clarified the statement in the revised paper. As the reviewer correctly inferred, the final learning rate is 1/100 of the initial learning rate. This decay factor applies proportionally to both parameter groups (the main model parameters starting at 10^{-5} and the bias parameters b_i starting at 10^{-3}), following standard annealing schedules (Loshchilov and Hutter, 2017). We have revised the text:

3.4 Training details

We use the AdamW optimizer (Loshchilov and Hutter, 2017) with ~~a~~ initial learning rate of 10^{-5} and a weight decay of 0.001. A higher initial learning rate of 10^{-3} is used for training the displacements b_i. ~~To progressively decrease the learning rate by a factor of 100, we~~ We utilize a cosine annealing schedule (Loshchilov and Hutter, 2016) to progressively decay the learning rate from its initial value η to a minimum value of $\eta/100$ over the course of training. Parameters are initialized using a standard

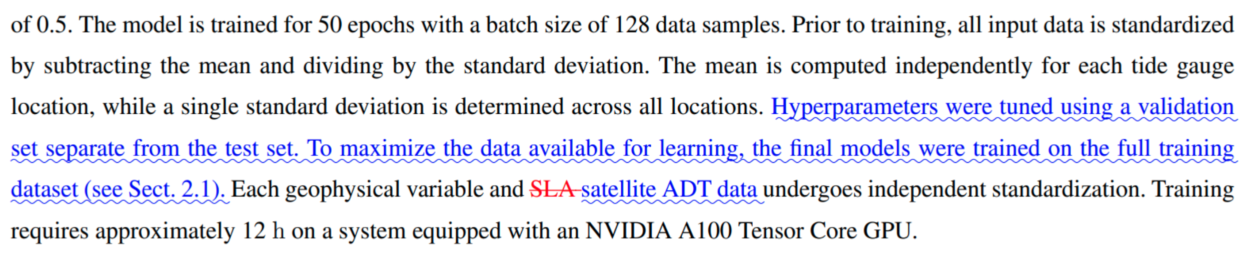
COMMENT 6:

Sec 2.5: More details required. Was a validation set used? Any early stopping conditions? How did you ensure overfitting did not occur?

RESPONSE 6:

While a separate validation set was used during the development phase to tune hyperparameters, the final models presented in the paper were trained on the full training dataset to maximize the historical data availability. We did not employ dynamic early stopping for the final training, we relied on a fixed schedule of 50 epochs with a learning rate scheduler. To prevent overfitting, the model architecture and training process incorporate weight decay, dropout layers within the Dense decoder, and a stochastic data augmentation strategy that randomly deactivates tide gauges, forcing the model to learn robust representations rather than memorizing specific sensor combinations.

We have expanded Section *Training details* to clarify our training protocol:



of 0.5. The model is trained for 50 epochs with a batch size of 128 data samples. Prior to training, all input data is standardized by subtracting the mean and dividing by the standard deviation. The mean is computed independently for each tide gauge location, while a single standard deviation is determined across all locations. Hyperparameters were tuned using a validation set separate from the test set. To maximize the data available for learning, the final models were trained on the full training dataset (see Sect. 2.1). Each geophysical variable and ~~SLA~~ satellite ADT data undergoes independent standardization. Training requires approximately 12 h on a system equipped with an NVIDIA A100 Tensor Core GPU.

COMMENT 7:

Sec 3.1: This model description subsection seems better suited for the methods than the results.

RESPONSE 7:

As suggested, we have reorganized the manuscript structure. The NEMO model description is now located in Section 2.4, within the Data and experimental setup section, separate from the section describing the model architecture.