Forecasting Sea Surface Temperature from Satellite Images with Graph Neural Networks

Giovanny A. Cuervo-Londoño[1 [0000−0002−8368−7324]], Javier Sánchez[2 [0000−0001−8514−4350]]✉, and Ángel Rodríguez-Santana[1 [0000−0003−1960−6777]]

[1] Instituto Universitario de Investigación en Acuicultura Sostenible y Ecosistemas Marinos (ECOAQUA)
[2] Instituto Universitario de Cibernética, Empresas y Sociedad (IUCES)
University of Las Palmas de Gran Canaria
35017 Las Palmas de Gran Canaria, Spain
{giovanny.cuervo101}@alu.ulpgc.es
{jsanchez,angel.santana}@ulpgc.es

Abstract. Predicting the evolution of sea surface temperature (SST) is essential for applications in weather forecasting, maritime transport, and fisheries. Traditional ocean forecasting methods rely on physics-based numerical models, which face challenges such as data gaps, assimilation difficulties, and computational inefficiencies. Recent advances in Graph Neural Networks (GNNs) have shown promise in improving prediction accuracy and efficiency. In this work, we adapt a GNN model, initially designed for atmospheric forecasting, to oceanographic applications. We focus on the Canary Islands and the northwest African shore regions characterized by strong mesoscale dynamics. Our approach introduces a spatially masked loss function to address ocean-specific challenges, such as spatial discontinuities and observational data sparsity. We train our model using the L4 SST satellite images dataset from Copernicus Marine Service and compare its performance with state-of-the-art ConvLSTM-based models. Our results indicate that the adapted GNN model effectively captures mesoscale structures and outperforms ConvLSTM in both computational efficiency and accuracy. These findings suggest that graph-based deep learning approaches can overcome key limitations of current oceanographic models and provide a more flexible and scalable solution for forecasting oceanographic variables from satellite images.

Keywords: Graph neural network · Deep learning · Remote sensing · Forecasting · Oceanography.

1 Introduction

Forecasting the evolution of oceanographic data obtained from satellite images is necessary for tasks related to weather prediction, maritime transport, and the fishery industry [2], among others. It relies heavily on forecasting mesoscale processes due to their environmental and economic impacts. These processes,

which give rise to distinct structures, also influence mean currents and transport key ocean properties. Despite its importance, predicting mesoscale processes remains a challenging task [10].

Traditional ocean forecasting systems rely on numerical models that use physics-based equations, although they have several limitations. Gaps in observational data and ongoing challenges with current data assimilation techniques make it difficult to get a complete picture of the ocean. Additionally, these models do not fully leverage historical data or take advantage of modern hardware like GPUs, making them less efficient.

Deep learning models have recently emerged in global atmospheric forecasting, considerably improving the performance of numerical systems. There have appeared numerous models for predicting the state of the atmosphere, such as Pangu Weather [3], GraphCast [9], Aurora [4], NeuralGCM [8], or Gencast [11], which rely on foundational models, graph neural architectures, or diffusion models. These have considerably reduced inference times and computational costs.

These models rely on high-quality data but struggle with inconsistent, sparse, or noisy datasets, which can limit their accuracy. They also suffer from spectral bias in the long run, leading to numerical instability or unrealistic predictions. While these models work well for atmospheric systems, applying them to ocean data is more challenging because of the atmospheric interference and landmasses. Nevertheless, some models have recently appeared for predicting the ocean dynamics, such as Xihe [13], designed for global predictions, or SeaCast [7] and OceanNet [5], with a focus on regional forecasting.

Forecasting SST in coastal regions like the Canary Islands and northwestern Africa is crucial due to its significant implications for marine ecology, weather prediction, and fisheries. While prior deep learning approaches have shown strong results in atmospheric forecasting, regional oceanographic forecasting remains relatively unexplored.

In this work, we adapt a graph neural network [9], initially designed for atmospheric systems, to oceanographic forecasting. To tailor our model for this task, we adapt the underlying graph to a subregional domain and introduce a spatially masked loss function optimization that targets oceanic regions and prevents land-related artifacts during training. This approach differs from conventional methods, which are typically applied before training. By restricting loss computation to oceanic zones, we explore whether it can mitigate spectral bias by forcing the model to learn in selected marine areas during training.

Our study focuses on the Canary Islands and the northwest African shore subregion, which presents several challenges due to the oceanic and coastal interactions [12]. Traditional methods struggle to represent these interactions due to the highly nonlinear processes caused by strong mesoscale, upwelling fronts, and complex eddy fields. We use the L4 satellite-derived SST dataset [6] from the Copernicus Marine Service (CMEMS) for the Atlantic Ocean around Iberia, Biscay, and Ireland (IBI), limited to the Canary Islands subdomain. This dataset comprises nearly 40 years of daily SST satellite images from 1982 to 2020, with a resolution of 5.55 km per pixel.

By comparing our model to a state-of-the-art model, such as ConvLSTM [14], we assess the benefits and drawbacks of GNNs and how they can better capture these unresolved dynamics and offer greater flexibility. Additionally, we evaluate the model's sensitivity to mesoscale processes of high-dynamical complexity areas, such as frontal zones near the islands and African capes. The experimental results show that our model surpasses ConvLSTMs in computational efficiency and accuracy. We evaluate its performance in short- and long-range predictions, emphasizing its spatial behavior in our study area.

Our work introduces three main contributions: i) it adapts a global GNN for subregional SST forecasting; ii) it introduces a spatially masked loss function to target only oceanic areas, avoiding land artifacts; iii) the experiments evaluate mesh resolution trade-offs concerning both accuracy and computational cost.

Section 2 explains the dataset and the study area of this work. Section 3 details the architecture of our graph neural network. The experimental results (Section 5) evaluate and compare the model's performance with ConvLSTM. Finally, Section 6 discusses the contributions and limitations of this work and proposes future research directions.

2 Dataset and Study Area

This study focuses on the Canary Islands and the Moroccan subregion, which extends from 21°S to 33°N; see Fig. 1. This region has strong upwelling due to wind and seabed shape, bringing cold water and nutrients to the surface. The capes push coastal currents offshore, forming filaments that move water into the ocean. Irregular seabed modifies currents, creating cyclonic eddies and boosting marine life.

We use the L4 SST dataset [6] from the Copernicus Marine Service (CMEMS) for the Atlantic Ocean around Iberia, Biscay, Ireland (IBI), and the northwestern European shelf domain; see Fig. 1 on the left. It comprises nearly 40 years of SST satellite images from 1982 to 2020, providing a gap-free daily SST estimate, using multiple satellite observations and ensuring temporal consistency. The image resolution is about 0.05 degrees ($\approx 5.55\,km$ per pixel).

Our study area is a subdomain within the IBI region, ranging from 19.55° to 34.525° latitude and -20.97° to -5.975° longitude, covering an area of approximately $2,462,475\,km^2$; see Fig. 1 on the right. This region is represented by a grid of 300×300 cells. The temporal range of the data used spans from January 1, 1982, to December 31, 2020, corresponding to a total of $14,245$ frames (daily images) and a storage size of 10.25 GB.

We preprocessed the dataset to fill in missing values by first generating a binary land-sea mask, smoothed using a Gaussian filter. Then, we replaced the missing values in the continent with the average SST to maintain data continuity. This information is used as boundary conditions for our method. Figure 2 shows several samples of the L4 dataset in 2020.

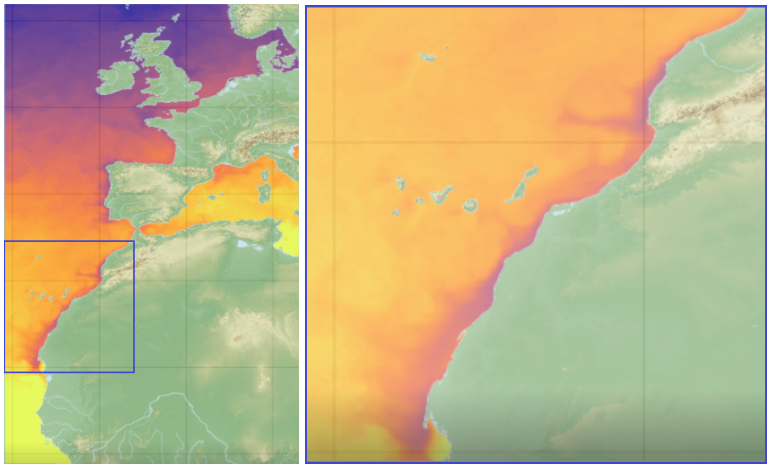

Fig. 1. Iberia, Biscay, and Ireland (IBI) region represented in the L4 dataset on the left and the subregion we use in this work on the right, comprising the Canary Islands, Madeira, and the northwestern African shore.

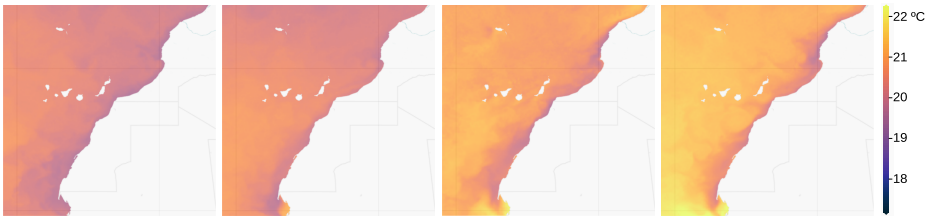

Fig. 2. Several samples of the L4 dataset. These images are examples of the SST on different days of 2020, corresponding to January, April, July, and October, respectively. Colors represent the sea surface temperature in ^{o}C, with bright colors representing higher temperatures.

3 Forecasting with Graph Neural Networks

Graph Neural Networks (GNN) are an efficient mechanism to model dependencies based on the geometry of the problem. This section explains our adaptation of a GNN architecture, based on the GraphCast [9] model, for regional oceanographic forecasting. On the one hand, we limit the number of variables to the SST and replace the original graph with a planar mesh, which reduces the complexity of the model and increases the computational efficiency. On the other hand, we adjust the latent size of the features in each node to reduce memory requirements.

The input variable is a spatiotemporal volume, represented as $\mathbf{x} : \mathbb{R}^3 \rightarrow \mathbb{R}$. x_i^t is a scalar value, with i standing for node $v_i \in \mathcal{V}^g$, and t the time instant.

Our GNN is defined as an autoregressive model,

$$\hat{\mathbf{x}}^{t+1} = f(\mathbf{x}^t, \mathbf{x}^{t-1}), \tag{1}$$

where $\hat{\mathbf{x}}^{t+1}$ is estimated from two previous values and can be fed into the model to predict future states. The graph, $\mathcal{G}(\mathcal{V}^g, \mathcal{V}^m, \mathcal{E}^m, \mathcal{E}^{g2m}, \mathcal{E}^{m2g})$, is composed of grid nodes, $\mathcal{V}^g$, mesh nodes, $\mathcal{V}^m$, bidirectional edges connecting mesh nodes, $\mathcal{E}^m$, and directed edges from grid to mesh nodes, $\mathcal{E}^{g2m}$, and vice-versa, $\mathcal{E}^{m2g}$.

Grid nodes are defined as $\mathbf{v}_i^g = [x_i^t, x_i^{t-1}, \mathbf{f}_i^{t-1}, \mathbf{f}_i^t, \mathbf{f}_i^{t+1}, \mathbf{c}_i]$, i.e., a combination of current and past SST values, temporal forcings, and static properties depending on spatial coordinates. Temporal forcings depend on the local time of day and the year progress, and constants on a binary land-sea mask and the node position.

Mesh nodes are defined as $\mathbf{v}_i^m = [\cos(\phi_i), \sin(\lambda_i), \cos(\lambda_i)]$, with ϕ_i and λ_i the longitude and latitude of node i, respectively. Mesh edges are defined as $\mathbf{e}_{s,r}^m = [\text{distance}(s, r), \mathbf{p}_s - \mathbf{p}_r]$, i.e., the edge length and the difference between the spatial locations of the sender node, $mathbfp_s$, to the receiver node, $\mathbf{p}_r$. Unidirectional edges from the grid to the mesh, $\mathbf{e}_{s,r}^{g2m}$, and vice-versa, $\mathbf{e}_{s,r}^{m2g}$, are similarly defined. A mesh representation is shown in Fig. 3.

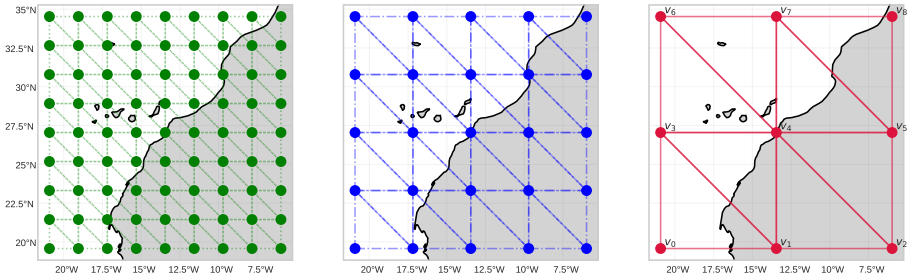

Fig. 3. The mesh nodes are organized into three distinct resolution levels: the finest level on the left (M^2, green) consists of 56 nodes, the intermediate level in the middle (M^1, blue) includes 16 nodes, and the coarsest level on the right (M^0, red) comprises 9 nodes.

The architecture of the GNN is composed of an *encoder* that converts the input data from the grid to the mesh, a *processor* that comprises multiple message-passing layers, and a *decoder* that converts the output data from the mesh to the grid; see Fig. 4. The *encoder* embeds the variables, $\mathbf{v}_i^g$, $\mathbf{v}_i^m$, $\mathbf{e}_{s,r}^m$, $\mathbf{e}_{s,r}^{g2m}$ and $\mathbf{e}_{s,r}^{m2g}$, into the latent space as

$$\tilde{\mathbf{v}}_i^g = \text{MLP}_{\mathcal{V}^g}(\mathbf{v}_i^g);\ \tilde{\mathbf{v}}_i^m = \text{MLP}_{\mathcal{V}^m}(\mathbf{v}_i^m) \tag{2}$$

$$\tilde{\mathbf{e}}_{s,r}^{g2m} = \text{MLP}_{\mathcal{E}^{g2m}}(\mathbf{e}_{s,r}^{g2m});\ \tilde{\mathbf{e}}_{s,r}^{m2g} = \text{MLP}_{\mathcal{E}^{m2g}}(\mathbf{e}_{s,r}^{m2g});\ \tilde{\mathbf{e}}_{s,r}^{m} = \text{MLP}_{\mathcal{E}^{m}}(\mathbf{e}_{s,r}^{m}); \tag{3}$$

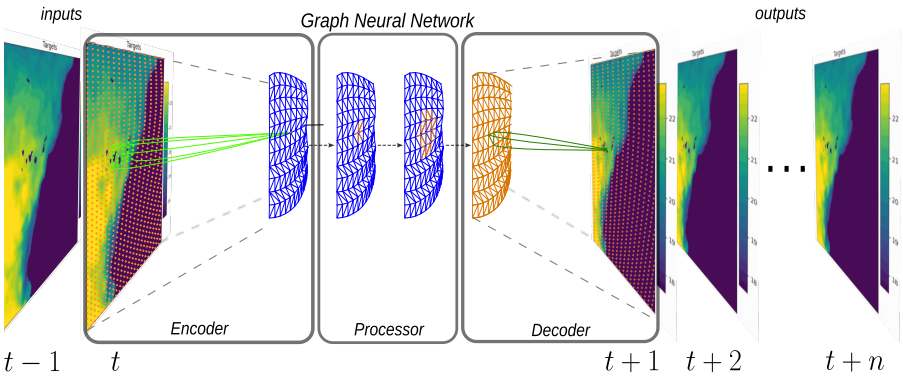

Fig. 4. Diagram of the *encoder-processor-decoder* architecture. The SST data is embedded in the mesh through the *encoder*, the *processor* transforms node features using multiple graph layers, and the *decoder* converts the internal mesh representation into a future SST forecast.

with MLP a multi-layer perceptron. The information is then transferred to the mesh using interaction networks (IN) [1], as follows:

$$\mathbf{d}\tilde{\mathbf{e}}_{s,r}^{g2m} = \mathrm{MLP}_{\mathcal{E}^{g2m}}([\tilde{\mathbf{e}}_{s,r}^{g2m}, \mathbf{s}^g, \mathbf{r}^m]);\ \mathbf{d}\tilde{\mathbf{v}}_i^m = \mathrm{MLP}_{\mathcal{V}^m}([\tilde{\mathbf{v}}_i^m, \sum_{s\in\mathcal{V}^g; r=\tilde{\mathbf{v}}_i^m} \tilde{\mathbf{e}}_{s,r}^{g2m}]), \quad (4)$$

with s the sender node, r the receiver node, and [.] the concatenation of multiple features. Grid nodes are also updated as

$$\mathbf{d}\tilde{\mathbf{v}}_i^g = \mathrm{MLP}_{\mathcal{V}^g}(\tilde{\mathbf{v}}_i^g), \quad (5)$$

In the final step of the *encoder*, we use residual connections as

$$\tilde{\mathbf{v}}_i^g \leftarrow \tilde{\mathbf{v}}_i^g + \mathbf{d}\tilde{\mathbf{v}}_i^g;\ \tilde{\mathbf{v}}_i^m \leftarrow \tilde{\mathbf{v}}_i^m + \mathbf{d}\tilde{\mathbf{v}}_i^m;\ \tilde{\mathbf{e}}_{s,r}^{g2m} \leftarrow \tilde{\mathbf{e}}_{s,r}^{g2m} + \mathbf{d}\tilde{\mathbf{e}}_{s,r}^{g2m}. \quad (6)$$

The *processor* contains various layers with the same mesh structure, whose parameters are calculated through message-passing. The edge and node features are updated as

$$\mathbf{d}\tilde{\mathbf{e}}_{s,r}^{m} = \mathrm{MLP}_{\mathcal{E}^m}([\tilde{\mathbf{e}}_{s,r}^{m}, \mathbf{s}^m, \mathbf{r}^m]);\ \mathbf{d}\tilde{\mathbf{v}}_i^m = \mathrm{MLP}_{\mathcal{V}^m}([\tilde{\mathbf{v}}_i^m, \sum_{s\in\mathcal{V}^m; r=\tilde{\mathbf{v}}_i^m} \tilde{\mathbf{e}}_{s,r}^{m}]), \quad (7)$$

with sender, s, and receiver, r, nodes in the mesh in this case, and they are updated with residual connections:

$$\tilde{\mathbf{v}}_i^m \leftarrow \tilde{\mathbf{v}}_i^m + \mathbf{d}\tilde{\mathbf{v}}_i^m;\ \tilde{\mathbf{e}}_{s,r}^{m} \leftarrow \tilde{\mathbf{e}}_{s,r}^{m} + \mathbf{d}\tilde{\mathbf{e}}_{s,r}^{m}. \quad (8)$$

The *decoder* performs the inverse operation to the *encoder*. The edge and node features are calculated with the following expressions:

$$\mathbf{d}\tilde{\mathbf{e}}_{s,r}^{m2g} = \mathrm{MLP}_{\mathcal{E}^{m2g}}([\tilde{\mathbf{e}}_{s,r}^{m2g}, \mathbf{s}^m, \mathbf{r}^g]);\ \mathbf{d}\tilde{\mathbf{v}}_i^g = \mathrm{MLP}_{\mathcal{V}^g}([\tilde{\mathbf{v}}_i^g, \sum_{s\in\mathcal{V}^g; r=\tilde{\mathbf{v}}_i^g} \tilde{\mathbf{de}}_{s,r}^{m2g}]). \quad (9)$$

A residual connection is used to update the information of the grid nodes coming from the embedding in the encoder $\tilde{\mathbf{v}}_i^g \leftarrow \tilde{\mathbf{v}}_i^g + \mathbf{d}\tilde{\mathbf{v}}_i^g$, and the output prediction is obtained with another MLP as $\hat{\mathbf{y}}^t = \text{MLP}_{\mathcal{V}^g}(\tilde{\mathbf{v}}_i^g)$. Finally, the forecasting is obtained through a residual connection as $\hat{\mathbf{x}}^{t+1} = \mathbf{x}^t + \hat{\mathbf{y}}^t$. All input variables are normalized to zero mean and unit variance. The size of the output layer in the last MLP, corresponding to the *decoder*, is one for the prediction of the value of the SST at each node.

4 Model Configuration and Training Details

The dataset was split by years from 1982 to 2012 in the training set, from 2013 to 2016 in the validation set, and from 2017 to 2020 in the test set, with approximately a ratio of 80%-10%-10%. The training phase followed the methodology explained in [9]. It involved 150 epochs with a half-cosine learning rate decay, starting at 10^{-3}. The model was trained to predict one lead time using three-time-instant samples. We used the AdamW optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.95$, $\epsilon = 10^{-8}$, and $\lambda = 0.1$. We also implemented gradient clipping when its norm was bigger than 32.

After an extensive hyperparameter search, we selected a configuration for the GNN model based on the model accuracy and our computational resources. We employed a three-level mesh as shown in Fig. 3. The processor module performed 6 message-passing steps within an 8-dimensional MLP latent space, with all MLPs containing a single hidden layer.

We used the Mean Square Error (MSE) to optimize the neural network, weighted by latitude to compensate for the spherical geometry of the grid cells. The loss function is given by:

$$\mathcal{L}_{MSE} = \frac{1}{|\mathbb{G}|} \sum_{v_i \in \mathbb{G}} w_\phi \left(\hat{x}_i^t - x_i^t\right)^2,$$

where $|\mathbb{G}|$ is the number of grid nodes, v is a node of the grid, and $w_\phi = \cos(\phi) / \left(\frac{1}{|\mathbb{G}|} \sum_{\mathbb{G}} \cos(\phi)\right)$ is a spatial varying weight that depends on the latitude ϕ. This weight decreases as we move towards the poles to account for smaller cell sizes.

We conducted experiments and hyperparameter search in a server equipped with 8 Quadro RTX 4000 GPUs (8 GB each). We used the Distributed Data-Parallel (DDP) approach for parallel training, with each GPU maintaining a copy of the model and processing independent batches, synchronizing gradients across devices through an all-reduce operation. This hybrid parallel-serial approach ensured efficient training while fully utilizing our hardware resources.

5 Results

In this section, we assess the performance of our model and compare it with state-of-the-art models. Table 1 shows the RMSE obtained for different hyperparameters. We test the mesh resolution using three configurations (M^2, M^4,

and M^6), various latent sizes, and several message passing steps in the processor. The batch size was 16 in all experiments. We observe that the RMSE is low for the three configurations; thus, it does not depend on the mesh resolution. The accuracy for M^2 and M^6 is similar, although the latter is more computationally demanding since the mesh contains exponentially more nodes. This is reasonable since the input data resolution is low, and M^2 seems to contain enough nodes to represent the input data.

Table 1. Comparison of the Root Mean Squared Error (RMSE) depending on the latent size, the number of resolution levels (M^2, M^4, and M^6), and the number of message-passing steps. Bold letters indicate the two most accurate solutions.

Configuration	**Latent-Size**	**Message-Passing**	**RMSE** (^oC)
GNN model (M^2)	2	2	0.0317
	4	6	0.0296
	8	7	**0.0294**
GNN model (M^4)	2	2	0.0323
	4	6	0.0301
	8	7	0.0299
GNN model (M^6)	2	2	0.0319
	4	6	**0.0293**

On the other hand, we observe an improvement when we increase the latent size and the number of message-passing steps. In this case, a latent size of eight seems the best value for an internal representation of the SST. In the work proposed in [9], the latent size was much bigger, according to the number of input variables. The difference between six and seven message-passing steps is low, so we retain the configuration with M^2, a latent size of eight, and six message-passing steps.

Figure 5 compares the predictions of our model with the ground truth values for five- to twenty-day forecasts. We observe that the error is low in short-time predictions and increases with longer predictions. As we observed in the 15- and 20-day forecasts, the method is very accurate in the deep sea and less accurate near the shore, where the temperature varies significantly.

Next, we compare our model with a state-of-the-art method based on the ConvLSTM architecture [14]. This model was trained with an input image of dimensions 256×256, and was optimized with the AdamW optimizer, a learning rate of 10^{-2}, a weight decay of 0.1, and 150 epochs.

Figure 6 shows the average RMSE of both models for 20-day forecasts. We calculated 20 predictions for each sample in the dataset and averaged the results for every lead time. The graph neural model maintains a lower error ratio in all lead times, providing more accurate solutions. On the other hand, Fig. 7 shows images of the spatial average of 5-, 10-, 15-, and 20-day forecasts for the whole

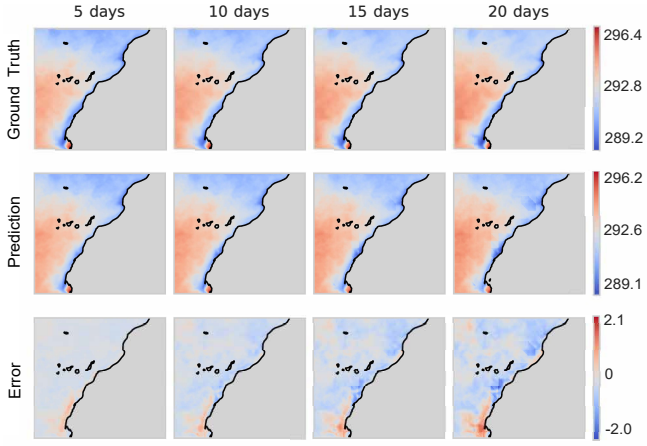

Fig. 5. Forecast comparison for several lead times. The initial date is April 5, 2020, with columns representing forecasts at 5-, 10-, 15-, and 20-day horizons, respectively. The top row shows the ground truth (GT) value for each day, the middle row shows model predictions for the same days, and the bottom row quantifies the error as the difference per pixel between the GT and the prediction.

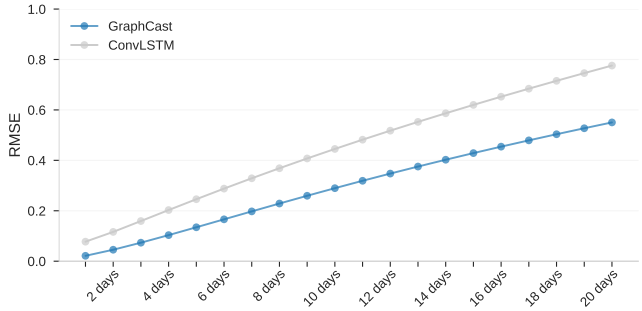

Fig. 6. Comparison of the RMSE values obtained by our model and the ConvLSTM model across a 20-day forecast horizon.

dataset. The graph model on the top clearly shows a lower average error in all the pixels of the images, meaning that the graph neural network can yield better predictions at any point in the domain. We also observe that the most significant errors are located near the coast.

An interesting observation is that, while the ConvLSTM produces smooth and continuous solutions, the graph neural network can estimate more discontinuous solutions. This allows the model to preserve small-scale variations in its predictions. We also note the presence of triangular shapes induced by the underlying mesh.

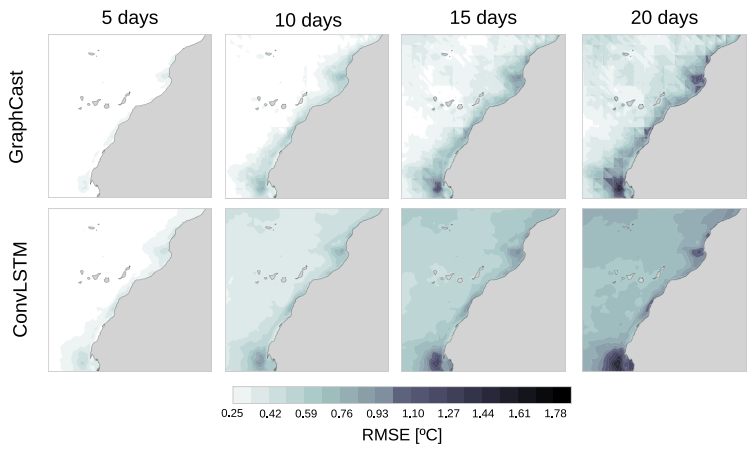

Fig. 7. Spatial distribution of the RMSE for 5-, 10-, 15-, and 20-day lead time forecasts. Results represent daily point-wise averages across the study domain. The graph model yields lower errors in most of the pixels. The highest errors are produced near the coast.

6 Conclusion

In this work, we adapted a GNN for global weather prediction to sub-regional oceanographic forecasting. We trained the model with the L4 dataset to analyze ocean dynamics in the Canary Islands and the northwest African shore. The spatial distribution of errors indicates that the model produces higher errors near the coast. These areas exhibit intense coastal interactions and complex bathymetric gradients. The lack of explicit atmospheric forcings, the bathymetry, and the coarse resolution of satellite data limit its ability to predict the SST in these regions. Our model consistently outperformed ConvLSTM in short- and long-range ocean forecasting, better capturing fine-scale ocean structures.

Some possible solutions to the triangular artifacts include refining the mesh, increasing connectivity between grid nodes, or using convolutional layers in the decoder. Addressing these limitations will be crucial for improving the model. Future work will also compare with other state-of-the-art methods.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Battaglia, P.W., Pascanu, R., Lai, M., Rezende, D., Kavukcuoglu, K.: Interaction networks for learning about objects, relations and physics. Tech. rep., Cornell University (2016), https://doi.org/10.48550/arXiv.1612.00222
2. Bell, M.J., Lefèbvre, M., Traon, P.Y.L., Smith, N., Wilmer-Becker, K.: GODAE: the global ocean data assimilation experiment. Oceanography **22**(3), 14–21 (2009), https://doi.org/10.5670/oceanog.2009.62

3. Bi, K., Xie, L., Zhang, H., Chen, X., Gu, X., Tian, Q.: Accurate medium-range global weather forecasting with 3D neural networks. Nature **619**(7970), 533–538 (2023), https://doi.org/10.1038/s41586-023-06185-3
4. Bodnar, C., Bruinsma, W.P., Lucic, A., Stanley, M., Brandstetter, J., Garvan, P., Riechert, M., Weyn, J., Dong, H., Vaughan, A., Gupta, J.K., Tambiratnam, K., Archibald, A., Heider, E., Welling, M., Turner, R.E., Perdikaris, P.: A foundation model of the atmosphere. Tech. rep., Cornell University (2024), https://doi.org/10.48550/arXiv.2405.13063
5. Chattopadhyay, A., Gray, M., Wu, T., Lowe, A.B., He, R.: OceanNet: a principled neural operator-based digital twin for regional oceans. Scientific Reports **14**(1), 21181 (Sep 2024), https://doi.org/10.1038/s41598-024-72145-0
6. CMEMS: European north west Shelf/Iberia Biscay Irish seas - high resolution L4 sea surface temperature reprocessed (2024), https://doi.org/10.48670/moi-00153
7. Holmberg, D., Clementi, E., Roos, T.: Regional ocean forecasting with hierarchical graph neural networks. arXiv e-prints arXiv:2410.11807 (Oct 2024), http://doi.org/10.48550/arXiv.2410.11807
8. Kochkov, D., Yuval, J., Langmore, I., Norgaard, P.C., Smith, J.A., Mooers, G., Klöwer, M., Lottes, J., Rasp, S., Düben, P.D., Hatfield, S., Battaglia, P.W., Sánchez-González, A., Willson, M., Brenner, M.P., Hoyer, S.: Neural general circulation models for weather and climate. Nature **632**, 1060 – 1066 (2023), https://doi.org/10.1038/s41586-024-07744-y
9. Lam, R., Sánchez-González, A., Willson, M., Wirnsberger, P., Fortunato, M., Alet, F., Ravuri, S., Ewalds, T., Eaton-Rosen, Z., Hu, W., Merose, A., Hoyer, S., Holland, G., Vinyals, O., Stott, J., Pritzel, A., Mohamed, S., Battaglia, P.: Learning skillful medium-range global weather forecasting. Science **382**(6677), 1416–1421 (2023), https://doi.org/10.1126/science.adi2336
10. Mourre, B., Aguiar, E., Juzà, M., Hernández-Lasheras, J., Reyes, E., Heslop, E., Escudier, R., Cutolo, E., Ruiz, S., Mason, E., Pascual, A., Tintoré, J.: Assessment of high-resolution regional ocean prediction systems using multi-platform observations: Illustrations in the Western Mediterranean sea. New Frontiers in Operational Oceanography (2018), https://doi.org/10.17125/gov2018.ch24
11. Price, I., Sánchez-González, A., Alet, F., Andersson, T.R., El-Kadi, A., Masters, D., Ewalds, T., Stott, J., Mohamed, S., Battaglia, P.W., Lam, R., Willson, M.: Probabilistic weather forecasting with machine learning. Nature **637**, 84 – 90 (2024), https://doi.org/10.1038/s41586-024-08252-9
12. Sangrà, P., Pascual, A., Ángel Rodríguez-Santana, Machín, F., Mason, E., McWilliams, J.C., Pelegrí, J.L., Dong, C., Rubio, A., Arístegui, J., Ángeles Marrero-Díaz, Hernández-Guerra, A., Martínez-Marrero, A., Auladell, M.: The Canary eddy corridor: A major pathway for long-lived eddies in the subtropical North Atlantic. Deep Sea Research Part I: Oceanographic Research Papers **56**(12), 2100–2114 (2009), https://doi.org/10.1016/j.dsr.2009.08.008
13. Wang, X., Wang, R., Hu, N., Wang, P., Huo, P., Wang, G., Wang, H., Wang, S., Zhu, J., Xu, J., Yin, J., Bao, S., Luo, C., Zu, Z., Han, Y., Zhang, W., Ren, K., Deng, K., Song, J.: XiHe: A data-driven model for global ocean eddy-resolving forecasting. Tech. Rep. arXiv:2402.02995, Cornell University (Feb 2024), http:/doi.org/10.48550/arXiv.2402.02995
14. Yang, J., Zhang, T., Zhang, J., Lin, X., Wang, H., Feng, T.: A ConvLSTM nearshore water level prediction model with integrated attention mechanism. Frontiers in Marine Science **11** (2024), http://doi.org/10.3389/fmars.2024.1470320