




APPLICATION PAPER 

The challenge of land in a neural network ocean model

Rachel Furner^{1,2} , Peter Haynes¹, Dani C. Jones^{2,3}, Dave Munday², Brooks Paige⁴ and Emily Shuckburgh⁵

¹Department of Applied Mathematics and Theoretical Physics, University of Cambridge, Cambridge, United Kingdom

²Polar Oceans Team, British Antarctic Survey, Cambridge, United Kingdom

³School for Environment and Sustainability, Cooperative Institute for Great Lakes Research, University of Michigan, Ann Arbor, MI, USA

⁴University College London, London, United Kingdom

⁵Department of Computer Science and Technology, University of Cambridge, Cambridge, United Kingdom

Corresponding author: Rachel Furner; Email: raf59@cam.ac.uk

Received: 14 March 2024; **Revised:** 03 October 2024; **Accepted:** 16 October 2024


Keywords: data science; machine learning; oceanography; data-driven; forecasting; coast

Abstract

Machine learning (ML) techniques have emerged as a powerful tool for predicting weather and climate systems. However, much of the progress to date focuses on predicting the short-term evolution of the atmosphere. Here, we look at the potential for ML methodology to predict the evolution of the ocean. The presence of land in the domain is a key difference between ocean modeling and previous work looking at atmospheric modeling. Here, we look to train a convolutional neural network (CNN) to emulate a process-based General Circulation Model (GCM) of the ocean, in a configuration which contains land. We assess performance on predictions over the entire domain and near to the land (coastal points). Our results show that the CNN replicates the underlying GCM well when assessed over the entire domain. RMS errors over the test dataset are low in comparison to the signal being predicted, and the CNN model gives an order of magnitude improvement over a persistence forecast. When we partition the domain into near land and the ocean interior and assess performance over these two regions, we see that the model performs notably worse over the near land region. Near land, RMS scores are comparable to those from a simple persistence forecast. Our results indicate that ocean interaction with land is something the network struggles with and highlight that this is may be an area where advanced ML techniques specifically designed for, or adapted for, the geosciences could bring further benefits.

Impact Statement

Process-based models provide excellent tools to understand and predict the earth system but are computationally costly. Machine learning (ML) techniques are being leveraged to provide computationally cheaper models which run in a fraction of the time and have been shown to predict atmospheric weather at accuracies comparable to leading operational forecast systems. To use ML-based models for ocean prediction, we need to consider the representation of land—grid points which are not themselves modeled but influence the flow. We develop a CNN ocean model and show that while the model has great predictive ability overall, near land the performance is poor. Simple methods used elsewhere for the representation of land are not sufficient. Our results indicate that this is an area where new ML techniques need to be developed, and these must be tested by assessing performance close to land in isolation.

 This research article was awarded Open Materials badges for transparent practices. See the Data Availability Statement for details.

1. Introduction

Accurate forecasting of weather and climate has huge societal benefits. However, the systems used for these predictions are computationally expensive. Machine learning (ML) offers a new methodology with the potential for faster and computationally cheaper forecast systems. Dueben and Bauer (2018) was one of the first papers to assess the potential for ML-based weather prediction systems, training a deep neural network to learn the evolution of a single 2D variable. Since this, there has been a growing interest in this field. There are now numerous papers looking at training ML methods to learn the evolution of variables from weather or climate observation datasets or from the output of General Circulation Models (GCMs). Various network architectures have been applied to the problem, such as convolutional neural networks (CNNs) (Scher, 2018; Scher and Messori, 2019; Weyn et al., 2019, 2020), reservoir computing (Arcomano et al., 2020), and Residual Network architecture (Clare et al., 2021; Rasp and Thuerey, 2021).

More recently there have been substantial improvements in performance of these models (based on RMS errors), with ML-based methods providing forecasts which are competitive with operational Numerical Weather Prediction (NWP) systems. Specifically Pathak et al. (2022) used a Fourier neural operator; Keisler (2022), Bi et al. (2023) and Lam et al. (2023) used graph neural networks; and Chen et al. (2023a) used a transformer-based architecture. All achieve excellent performance in comparison to the European Centre for Medium-range Weather Forecasting (ECMWF) Integrated Forecasting System (IFS), often bettering this system. Further developments in this area look to address the ensemble prediction problem. Price et al. (2023) developed Gencast, an ensemble prediction model, and Chen et al. (2023b) presented a transformer-based cascaded architecture. Compared against the ECMWF IFS ensemble system both show huge skill, again often bettering the operational model.

There has also been interest in using data-driven models to address a wider array of problems. Watson-Parris et al. (2022) proposed a benchmark for climate forecasting—highlighting the very different requirements of this task in comparison to weather forecasting. A more general transformer architecture is developed in the study by Nguyen et al. (2023), designed as a foundation model which can be used for a wide range of weather and climate problems.

Despite the growing interest in this area, examples to date predominantly focus on atmospheric forecasting, with few examples of the use of ML for ocean prediction. Furner et al. (2022) used a very simple regression model to learn the dynamics of output from a GCM of a highly idealized ocean configuration, with a focus on analyzing the underlying sensitivity of the regressor. The studies by Wang et al. (2024) and Xiong et al. (2023) are the first examples we are aware of that attempt to model the entire global ocean with ML techniques. Xiong et al. (2023) developed AI-GOMS, a Fourier-based autoencoder, trained to predict ocean variables 1 day ahead. They use this to make autoregressive predictions up to 30 days ahead and compare this to FourCastNet (Pathak et al., 2022), an atmospheric ML model. Results show AI-GOMS outperforms FourCastNet; however, it should be noted that FourCastNet has been applied to the ocean forecast problem with no adjustments to adapt it to ocean modeling. Wang et al. (2024) developed XiHe, a $\frac{1}{12}^\circ$ global ocean model, using a hierarchical transformer framework, along with specific land–ocean mask mechanism. This is trained to directly predict for lead times as far out as 60 days and compared to operational ocean forecast models. XiHe outperforms these operational models on the assessment metrics, even as far out as 60 days.

While the dynamics of the ocean and atmosphere are similar in many ways, predicting the ocean raises different challenges to atmospheric modeling. Here, we focus on the impact of including lateral boundaries in the computational domain. Unlike atmospheric models, the oceans are laterally bounded by land, and here, we consider the treatment of, and impact of, land. While we are only interested in the dynamics of the ocean, this is impacted by the land, meaning the domain contains points that are not included in predictions, yet still influence predictions. By contrast, atmospheric models tend to use a terrain following coordinate system, which removes the technical issues of dealing with lateral boundaries.

This research looks to address the impact of land when modeling an ocean using CNNs—a technique which has been used for atmospheric models with much success. We run the dynamic model MITgcm (Marshall et al., 2004) in an idealized ocean channel configuration with two land peninsulas, and with

simple atmospheric forcing, to create a dataset. Following the work of Scher (2018) and Weyn et al. (2019), we then train a CNN to emulate the evolution of the MITgcm dataset. We assess results both over the entire domain and by separating the domain into near land and ocean interior regions. While Wang et al. (2024) specifically discussed the mechanism used for including land in the domain, neither they nor Xiong et al. (2023) assessed the impact by looking specifically at results near land as we do here.

This paper begins in section 2 by discussing the MITgcm simulator configuration and resultant dataset and the neural network architecture and training process. To avoid confusion, we refer to MITgcm as the simulator, and references to “the model” refer to our CNN. Section 3 presented the performance of the neural network. We discussed these results in more context in section 4. The paper concludes with section 5.

2. Method

2.1. MITgcm simulation

We run a channel configuration of MITgcm similar to that used in the study by Munday et al. (2015) to create training, validation, and test datasets. The configuration runs with 10 km horizontal grid spacing, with 240 grid points in the x -direction and 104 grid points in the y -direction (the domain is 2400 km by 1040 km, which is reduced from that used in the study by Munday et al. (2015) in order to reduce computation time in training our CNN). There is a flat bottomed bathymetry, of depth 5 km, with 38 unevenly spaced z -levels. The domain is bounded by land on the Northern and Southern boundaries and has two peninsulas extending North and South, as shown in Figure 1. At land boundaries, the boundary condition of no normal flow is imposed. This enforces the insulating condition of no heat or mass transport through land boundaries (i.e. temperature and sea surface height [SSH] cannot change due to interactions with land). The domain has periodic boundary conditions on the Eastern and Western boundaries.

Temperature is initialized with a linear meridional gradient at the surface, with maximum temperature 7°C in the North and minimum 0°C at the Southern boundary, and exponential decay through depth, with a small amount of white Gaussian noise added. Surface restoring is applied for temperature, with a restoring timescale of 10 days. A sinusoidal jet of wind is applied just North of the center of the domain

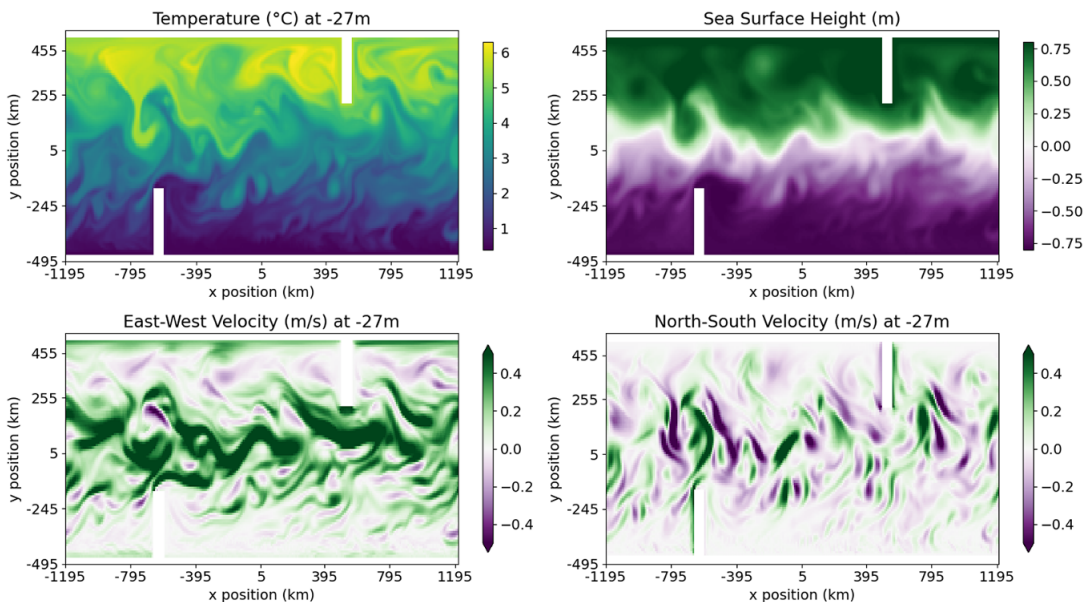


Figure 1. Example 12-hour averaged MITgcm fields for temperature, SSH, and Eastward and Northward velocity components.

with a maximum wind stress of 0.2 Nm^{-2} . The simulator is run with constant salinity of 35 PSU, meaning it is effectively not an active simulator variable and consequently is not assessed. Further details about the configuration can be found in the study by Munday et al. (2015). The main differences between the configuration used here and that described in the study by Munday et al. (2015) are that we omit the enhanced vertical diffusion at the Northern boundary and reduce the size of the channel.

The configuration is an idealized version of a Southern Ocean Channel configuration, with temperatures and wind forcing comparable to that found in the Southern Ocean, and the inclusion of land peninsulas to disrupt the flow. While being idealized in many ways, this configuration gives rise to very realistic and complicated dynamics, including a front in SSH and temperature, the formation of eddies along the front, and jets in the currents.

The simulator is run from rest for 100 years. The first 50 years of data is considered to cover a period during which the simulator is spinning up (adjusting from the initial conditions to more realistic ocean dynamics) and so is discarded. The second 50 years is then used for training, testing, and validation datasets with a 75:15:10 split, discussed below. The GCM runs with an internal time step of 10 minutes, and we output 12-hour averaged fields for temperature, SSH, and U (Eastward) and V (Northward) components of velocity.

Sample fields are shown in Figure 1. We see there is a complex and detailed flow structure, with jets and eddies present, along with a notable front in both temperature and SSH. Mean fields and standard deviations calculated over the 37.5 year training dataset are shown in Figures 3 and 4. We see that the mean structure is dominated by meridional temperature and SSH gradients, and by an Eastward current in the center of the domain, driven by the wind forcing. The presence of the land peninsulas affect the mean flow, creating meridional structure in the velocity field, and causing detailed structure around land in the temperature fields. Standard deviations are highest in the center of the domain as expected; here, the Eastward flow is strongest and there is a sharp front in temperature and SSH giving rise to a high energy field with numerous eddies.

2.2. Dataset

Inputs to our neural network are the prognostic simulator variables, except the constant salinity. Namely 3D fields of temperature and Eastward and Northward components of velocity, and 2D fields of SSH, along with 2D fields of surface temperature flux and wind stress (the simulator boundary conditions). These variable fields are all provided at a given time step (as 12-hour averaged fields valid at that time), along with temporally fixed land–sea masks (see section 2.3 for details on this). The network output is the change over the next 12 hours in temperature, SSH, and Eastward and Northward components of velocity, that is, the change in all simulator variables except salinity. Input and output data is normalized by subtracting the mean and dividing by the range of each variable. This is done across all input variables and output variables separately, that is, the mean and range of the input values of temperature are calculated. These are calculated both spatially over the entire domain and over all samples in the training set. These are then used to normalize all input values of temperature. Then, this same process is applied to all other input variables and to all output variables. Appendix A shows the distribution of the dataset.

While the GCM runs with a 10-minute time step, we look at changes over a 12-hour period. Figure 2 gives an example of the changes seen over this time. This is because the internal time step of the GCM is based on a need for numerical stability, which given the underlying mechanisms within the GCM means running with short time steps. However, we focus on a prediction timescale over which changes in the ocean are of interest, here taken to be 12 hours.

MITgcm runs on an Arakawa C grid, which means variables have different spatial dimensions and positions. Specifically, Eastward and Northward components of velocities have an additional element in the x and y direction in comparison to the temperature and SSH variables. And the Eastward and Northward velocity points are located half a grid box to the East and South (respectively) of the temperature and SSH points. As most ML methods, including CNNs (which are used in this work), require all inputs to be on a consistently sized rectangular grid, we adapt the inputs for this. In the case of Northward velocity, the additional point is located in the land boundary to the North of the domain, and so

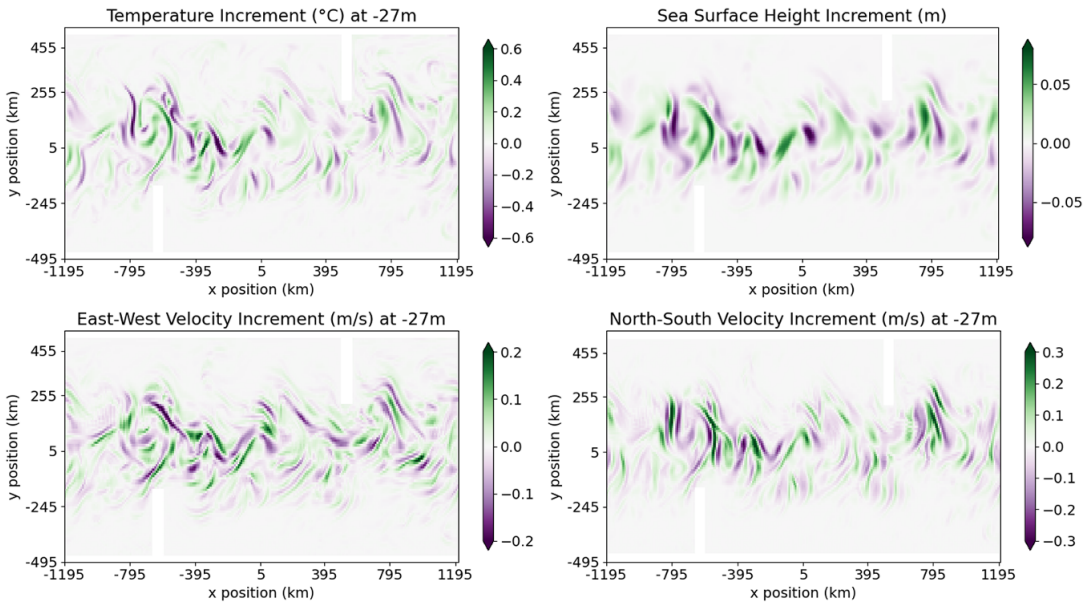


Figure 2. Example 12-hour increments to MITgcm fields for temperature, SSH, and Eastward and Northward velocity components.

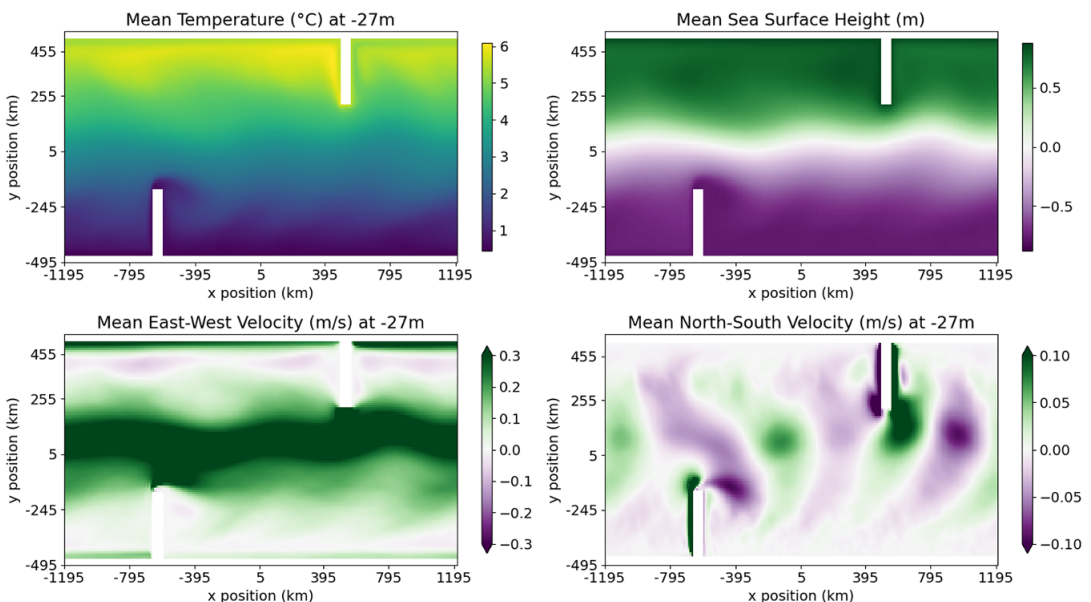


Figure 3. Mean MITgcm fields for temperature, SSH, and Eastward and Northward velocity components.

we simply exclude this row from our inputs. In the case of the Eastward velocities, the domain is periodic in the East–West direction, which means the first and last column of Eastward velocity data are identical. As such, we simply remove the additional (duplicated) column of Eastward velocity data. These changes mean that all variables are represented on grids of the same size and shape.

As is common practice, we subsample the data to reduce auto-correlation. We use a subsampling rate of 5 (i.e. we take every fifth pair of input–output fields), as we feel this gives a reasonable balance between

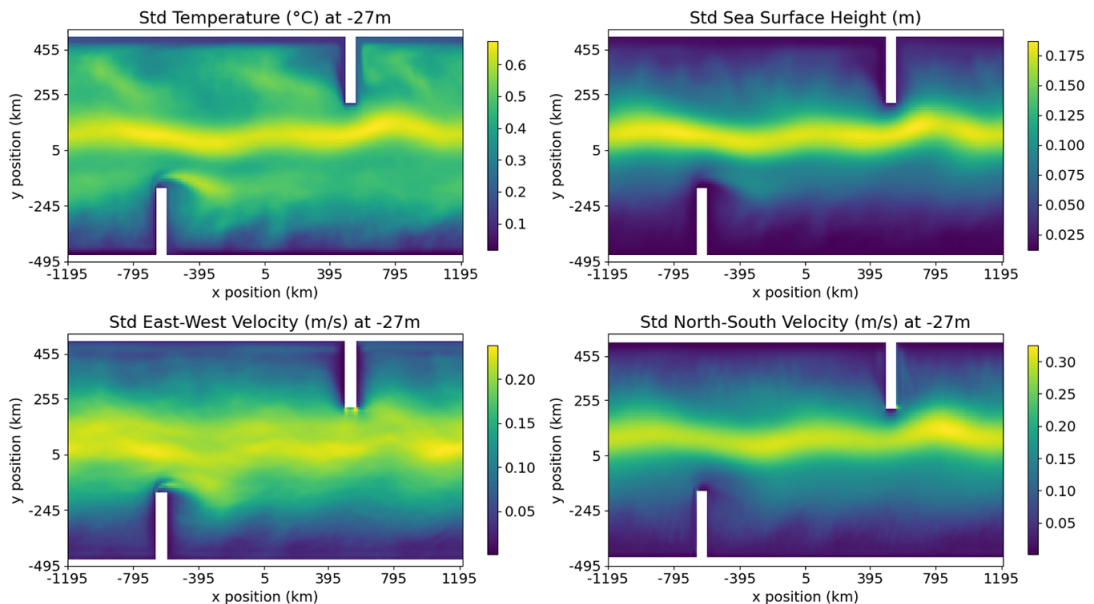


Figure 4. Standard deviation of temperature, SSH, and Eastward and Northward velocity components from MITgcm dataset.

reducing auto-correlation in the dataset and maintaining a large number of examples for training and validation. Our dataset is split to give 75% of the data as training data (used to train the neural network), 15% validation data (used to assess and compare different versions of the network during development), and a final 10% as test data (for reporting final statistics on). This is done sequentially so that the first 37.5 years are used for training, the next 7.5 years for validation, and the remaining 5 years for testing.

2.3. Treatment of land

To enable the network to understand the interactions between land and sea, the network is provided with the land–sea masks. These are temporally fixed 3D fields consisting of ones and zeros, with ones denoting grid locations that are ocean and zeros denoting locations that are land. There are three land–sea masks, one denoting the location of land for temperature and SSH points, one for Eastward velocities, and one for Northward velocities. The staggered grid used in MITgcm means that these different variables are located in slightly different locations, and so separate land–sea masks are needed for each grid. All three land–sea masks are given as inputs alongside the prognostic variables. Alongside this, physical inputs are all set to zero at land points prior to data being normalized. This methodology is similar to the way MITgcm sees land. We are not interested in how well the model performs over land, and so we mask the data before calculating the loss function, meaning losses are always exactly zero over land points.

Providing the land–sea masks means the network is aware of where land is, and so the network is theoretically able to capture the land–sea interactions present in the domain. This also allows for more realistic, irregular coastline shapes to be used. This approach is similar to that taken by Wang et al. (2024), with the exception that we are not harnessing the computational gains that Wang et al. (2024) obtained by removing calculations over some land regions due to the different architecture used here.

2.4. CNN setup

Following the work of Scher (2018), we train a U-Net style CNN (Ronneberger et al., 2015) to predict the evolution of the ocean simulation. A schematic of the network is shown in Figure 5. The network is made from combining various layers, all common to CNNs and U-Nets: convolution, batch norm, rectified

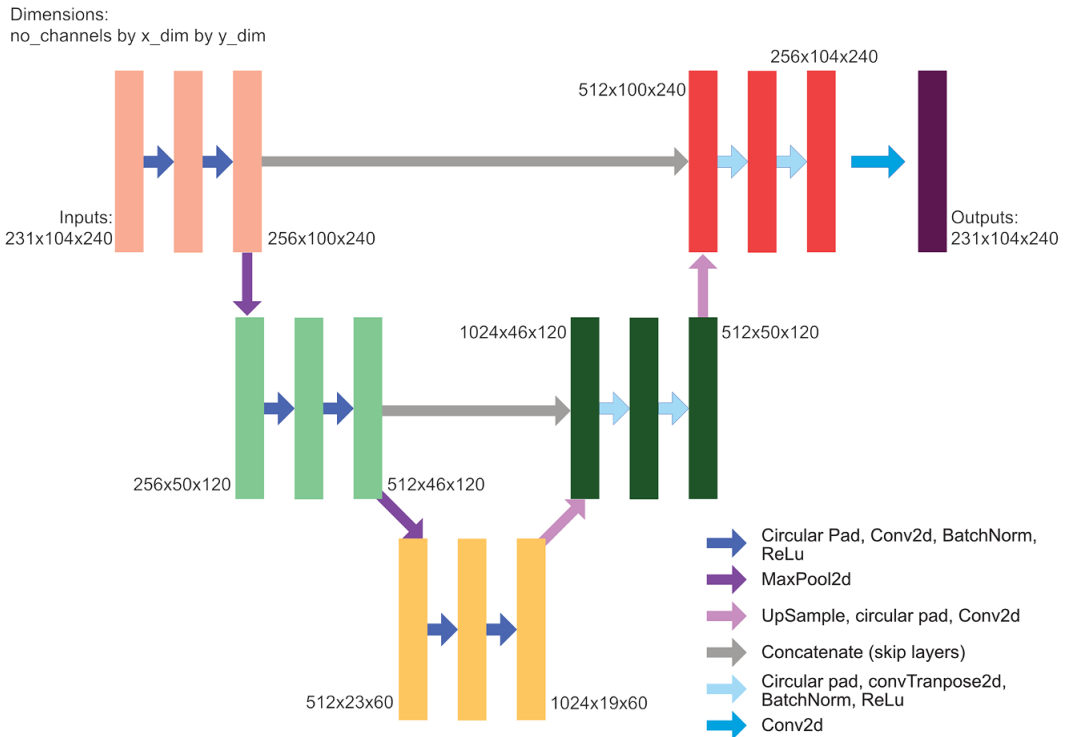


Figure 5. Schematic of the neural network used.

linear (ReLU), max pooling, upsampling, and a custom padding layer. Our custom padding applies circular padding at the East and West boundaries to replicate the periodic boundary conditions in the MITgcm simulation and no padding at the North and South boundaries. We use a 3-by-3 kernel size and a 2-by-2 max pooling window, both with a stride of 1. The number of channels increases from 155 input channels to 1024 channels at the deepest point of the network. This gives roughly 12.8 million trainable parameters. We run with the Adam optimizer and a constant learning rate of $3. \times 10^{-6}$. A variety of architectures were tested, with varying kernel size, varying pooling windows, and using a 3D structure rather than the current 2D structure, before selecting the current architecture based on performance over the validation set.

The CNN structure used here with 2D input fields means that the interactions are local in the horizontal and vertical, but global across variables and depth levels. Specifically, for any given point, the prediction is dependent upon the values of all variables (including the land–sea masks), over all depth levels, within a limited local horizontal region. Within the horizontal, predictions are most impacted by points directly next to the point being forecast, but the pooling used means that predictions are also impacted by points as far as 8 grid cells away horizontally.

We train for 200 epochs, meaning during training the network iterates through the entire training dataset 200 times, incrementally improving performance with each epoch. We use a loss function of mean squared error calculated over all variables at all points, with predictions masked so land points always give zero loss.

3. Results

3.1. Results over entire domain

Figure 6 shows the training evolution of RMS errors for predictions from the training and validation datasets along with training errors separated out for each variable. We see that training is stable, and that by

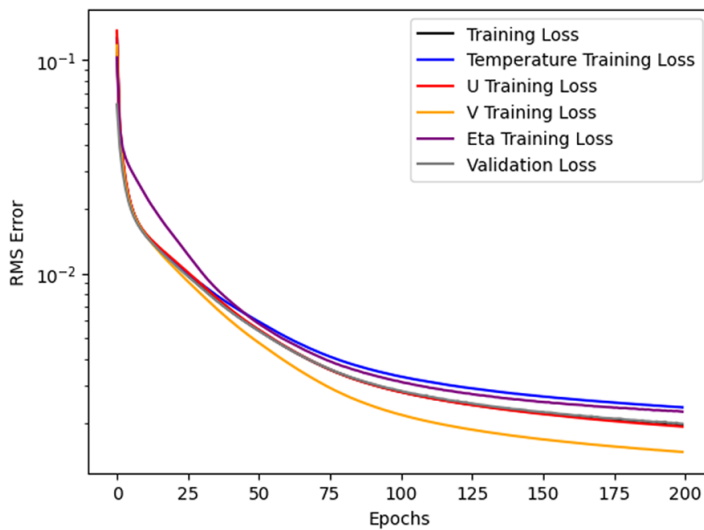


Figure 6. RMS error over training epochs, for the training and validation datasets, along with the training dataset separated by variable. Here, variables are all normalized using the mean and range, allowing fair comparison between variables.

200 epochs, training gains are minimal. After 200 epochs, RMS errors over the training and validation datasets are 1.96×10^{-3} and 1.98×10^{-3} , respectively. We can see from the figure that the network performs better over the velocity components than the temperature and SSH variables, with losses for the Northward component of velocity being particularly low.

Density scatter plots for each variable are shown in Figure 7, note we use a log scale. We plot the true increment against predicted increments for every grid point, including all depth levels, for in every sample in the test dataset. A persistence forecast (used as a baseline comparison for this work) would result in constant predictions of zero, that is, a line of data on $y = 0$. Here, we see that the performance is excellent across all variables, with very tight agreement between truth and predictions. Performance appears best for SSH, which shows very tight correlation.

To give an indication of how performance of the network varies spatially, Figure 8 shows RMS errors for the four predicted variables. For each 3D variable, this is shown at two vertical levels below the surface (this is the same depth shown in previous spatial plots). We see that the model performs well overall with errors low in comparison to the magnitude of changes being predicted. In the interior of the domain, RMS errors are less than 5% of the standard deviation of the dynamics, and near land this increases to maximums around 10–15% (not shown). All four variables show similar spatial patterns with errors largest in the center of the domain where the variability is highest and mean flow strongest, and large errors near to land, particularly around Northern peninsular highlighting that these regions are hardest to predict.

Figure 9 gives an example of the predicted increments from the neural network and the errors in these predictions. Again these are shown at two depth levels below the surface for 3D fields and are for the same sample (the same set of inputs), as shown in Figure 2. The predicted increments show detailed features around the center of the channel where we know the majority of change is occurring. We see that the predictions have a qualitatively realistic appearance, comparable to the real fields shown in Figure 2. Specifically, we see the resolution of features is maintained, and frequency and intensity of features are comparable. Errors for this particular example are low—at least an order of magnitude lower than the signal. The SSH errors show an interesting spatial bias, indicating that perhaps the model is misrepresenting a broad scale dynamical mode. For the velocity components, we see that similarly to Figure 8 the highest errors are located near land and in the center of the domain.

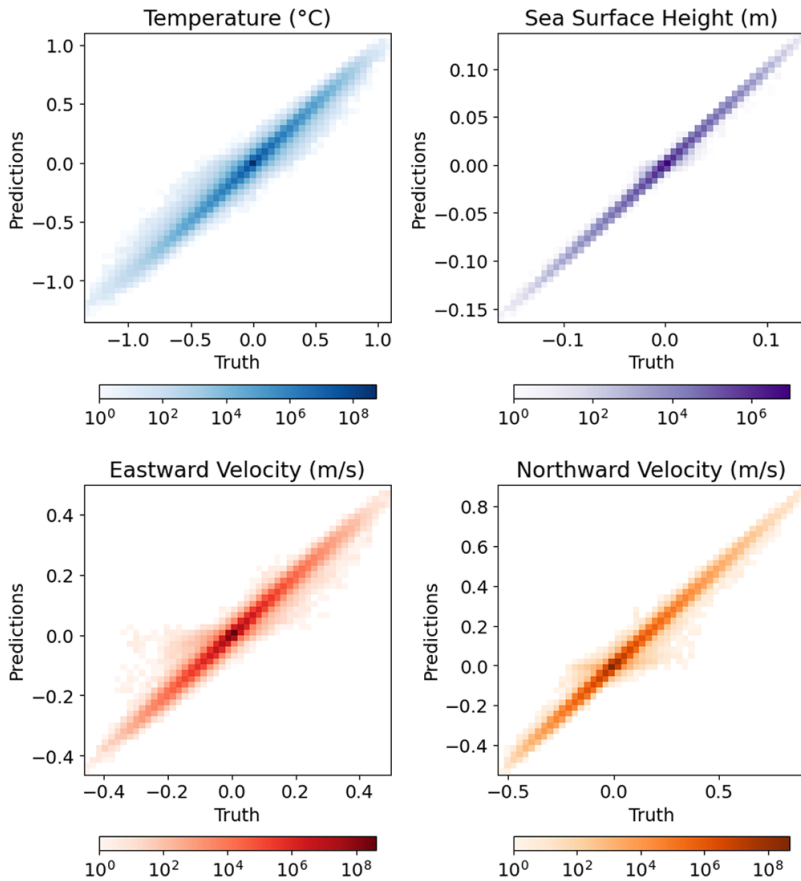


Figure 7. Log density scatter plots for predicted increments from the test dataset against true increments for temperature (top left), SSH (top right), Eastward component of velocity (bottom left), and Northward component of velocity (bottom right).

The errors in temperature prediction show large errors in the North-East corner of the domain, comparing this to Figure 1, we see this seems to be connected to a particular eddy. It should however be noted that this is a single randomly chosen sample, and so one should not draw too many conclusions from the particular errors seen here.

Table 1 shows RMS errors for each variable, averaged over the whole domain and over every sample in the test dataset. We show results from the neural network predictions and from a persistence forecast baseline. A persistence forecast is where we predict zero change—as the ocean changes slowly, a persistence forecast provides a good baseline for ocean models.

We also show the RMS errors from the network predictions normalized by the persistence forecast. Normalized results quantify the improvement the network provides over the persistence forecast and gives a meaningful way to compare performance across variables. Again, lower results indicate better performance. Scores of one imply that the neural network gives equivalent skill to the persistence forecast and scores lower (higher) than one indicate that the neural network has increased (reduced) skill over the persistence forecast. We see from the normalized scores that SSH predictions perform the best, followed by Northward velocity, and then Eastward velocity, with temperature performing the worst of all variables. Though even temperature (the worst performing variable) gives an order of magnitude improvement over the persistence forecast.

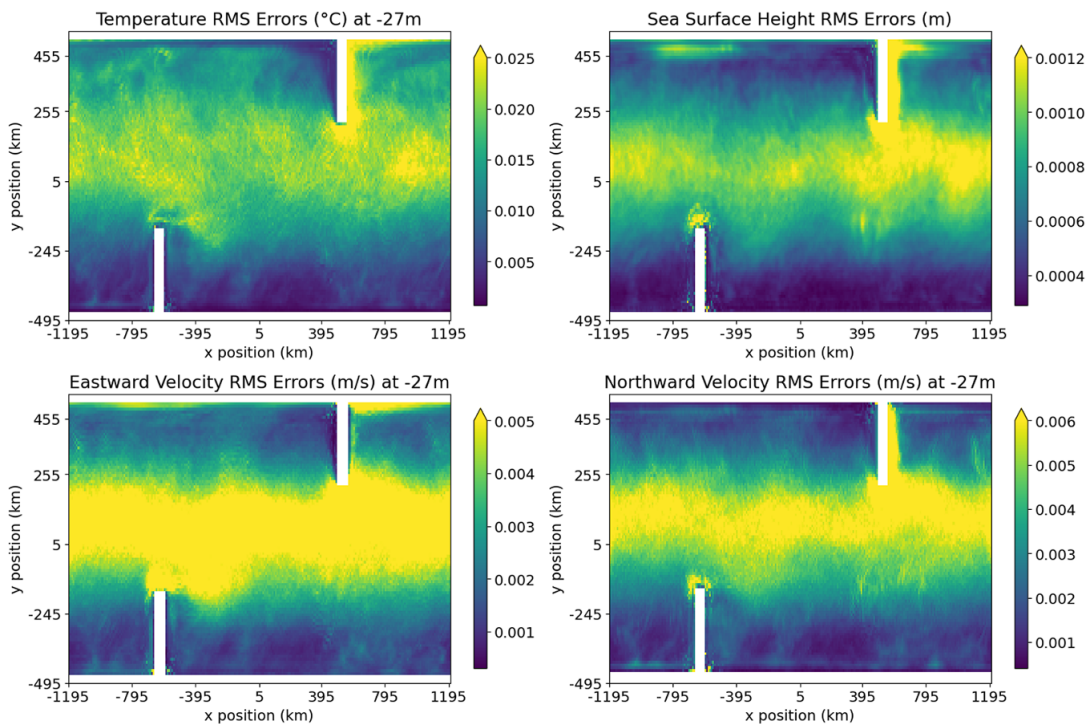


Figure 8. Spatial RMS errors averaged over all samples in the test dataset.

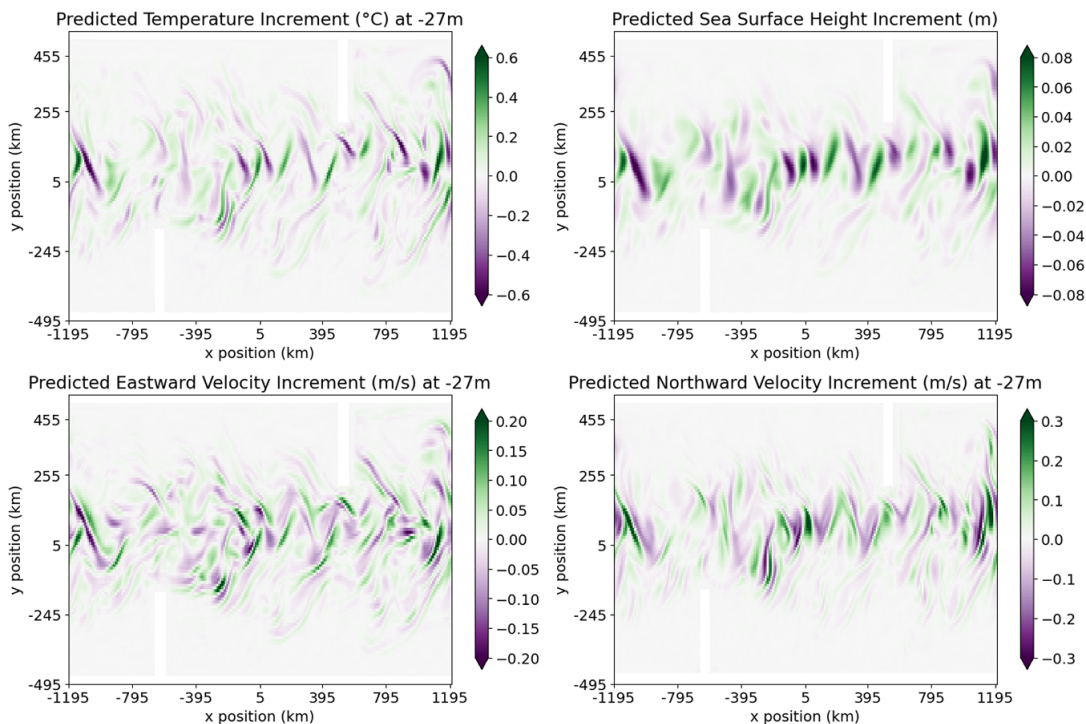


Figure 9. Example predicted 12 hour increments to MITgcm fields for temperature, SHH, and Eastward and Northward velocity components, from the test set.

Table 1. RMS errors for predictions from the neural network and from a persistence forecast, along with the neural network predictions normalized by the persistence forecast

	Network predictions	Persistence predictions	Network predictions Normalized by persistence
All variables	2.69×10^{-3}	1.93×10^{-2}	1.39×10^{-1}
Temperature	4.26×10^{-3}	2.9×10^{-2}	1.47×10^{-1}
SSH	7.85×10^{-4}	1.29×10^{-2}	6.11×10^{-2}
Eastward velocity	1.3×10^{-3}	9.13×10^{-3}	1.43×10^{-1}
Northward velocity	1.41×10^{-3}	1.41×10^{-2}	9.98×10^{-2}

Comparing to a persistence baseline, and especially normalizing scores relative to this, gives an indication as to whether RMS scores are low simply because very little is happening, or if RMS scores are low because, despite there being lots of activity, the network is capturing the dynamics well. This baseline allows us to separate the impact on RMS scores from the challenge of the configuration itself and the impact from the skill of the network. When comparing across different variables, the normalized skill score allows a fairer comparison, accounting for the different challenges posed by different variables.

3.2. Performance near land and over the ocean interior

Of particular interest to us is performance near land, and so we analyze that in more detail here. Figure 8 shows that all variables have some of their highest errors near land. However, the patterns are complex, with some near land regions performing better than others. Most variables have high errors around the Northern peninsula which, due to the slight asymmetry in the flow (see Figure 3), has more interaction with the peak current and fronts than the Southern peninsula. This interaction between the complex dynamics (jets, eddies, and ocean fronts), and the land is a very challenging area, and so it not surprising that we see the network struggle here. Elsewhere, performance varies across variables and specific coastlines, for example, SSH and Eastward velocity errors are slightly elevated along the Northern land boundary, but this does not occur along the Southern land boundary. We see that for most variables, performance along the Southern boundary is good, with low RMS errors. This is likely due to the underlying dynamics of the configuration; the Northern boundary is subject to greater vertical dynamics given the larger temperature stratification and has more significant boundary dynamics than the Southern boundary. The results imply that while the network struggles with land in regions with very complex dynamics, simpler land–sea interactions may be well modeled by the network in its current state.

We separate the domain into “coastal points,” that is, points which are within one grid cell (including diagonally) of land, and “ocean interior,” and analyze these areas separately to give greater insight into the impact of land on the model performance.

Figure 11 shows density scatter plots for each variable, for coastal points and for points in the ocean interior. We see a dramatic difference between the two plots. Performance of the network on coastal points is far worse than performance over the ocean interior for all variables, with much broader spread, and in some cases a failure to capture any meaningful correlation. Eastward velocity has the best performance near land, with the scatter plot showing some correlation, though the pattern is broader than that shown for the ocean interior, and there are issues with the model incorrectly predicting zero for some near zero values. Temperature performs particularly badly near land. The spread is very broad, and here, the model incorrectly predicts zero in a large number of cases. SSH and Northward velocity show similar performance near land, again patterns are very broad compared to performance over the ocean interior, with very little correlation. By contrast, for all variables, the performance within the ocean interior shows very strong correlations, as shown in Figure 7.

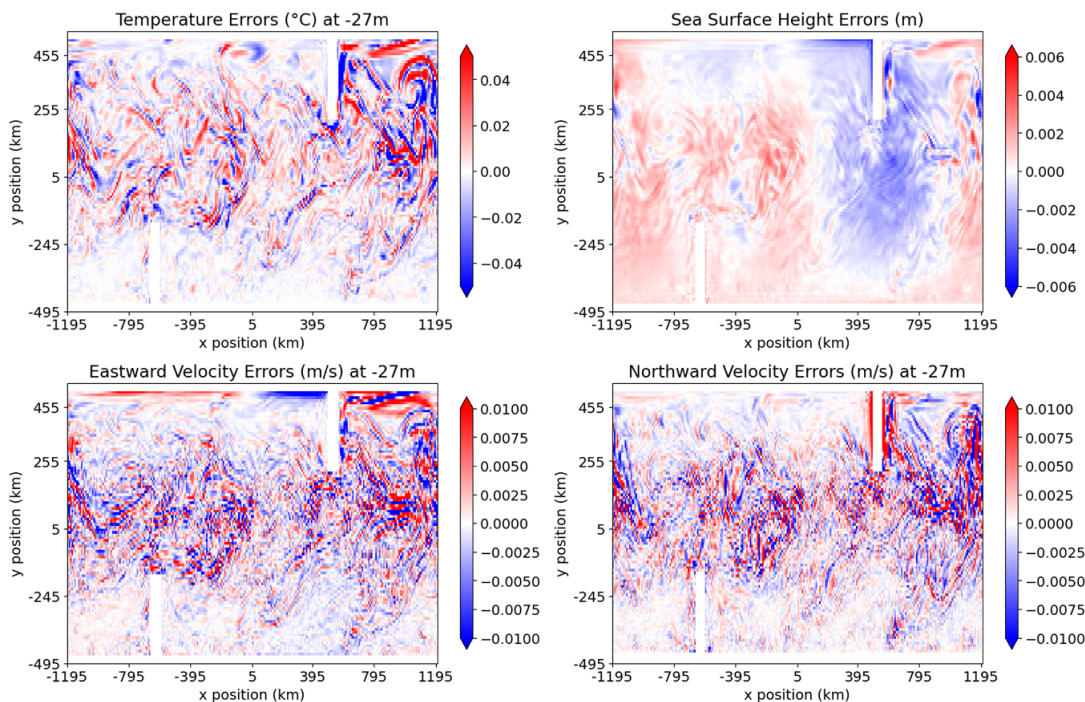


Figure 10. Errors in the above predicted 12-hour increments to MITgcm fields for temperature, SSH, and Eastward and Northward velocity components.

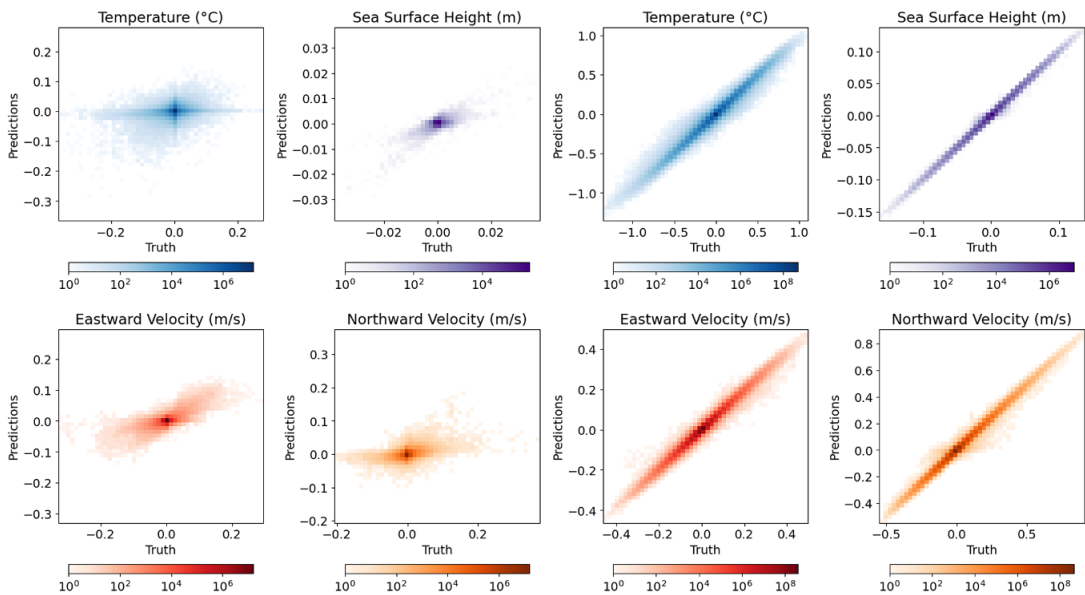


Figure 11. Log density scatter plots for coastal points for predicted increments from the test dataset against true increments for temperature (top left), SSH (top right), Eastward component of velocity (bottom left), and Northward component of velocity (bottom right), for coastal points (a) and ocean interior points (b).

Table 2. RMS errors and RMS errors normalized by persistence for the neural network predictions broken down into coastal points and the ocean interior

	RMS errors		Normalized RMS errors	
	Coastal	Ocean interior	Coastal	Ocean interior
All variables	3.73×10^{-4}	2.69×10^{-3}	1.05	1.38×10^{-1}
Temperature	3.42×10^{-3}	4.28×10^{-3}	1.11	1.46×10^{-1}
SSH	8.09×10^{-4}	7.85×10^{-4}	9.06×10^{-1}	6.03×10^{-2}
Eastward velocity	1.66×10^{-3}	1.29×10^{-3}	$9. \times 10^{-1}$	1.40×10^{-1}
Northward velocity	1.72×10^{-3}	1.4×10^{-3}	1.03	9.8×10^{-2}

Table 2 shows the RMS errors for each variable broken down into coastal points and ocean interior points. When looking directly at RMS errors, we see that for most variables, RMS errors are lower for the coastal region than for the ocean interior. However, comparing RMS errors in this way conflates impacts from model skill with impacts from the difficulty of the forecast problem. We would expect that, as values at these coastal points vary less than those in the ocean interior, predicting values here may naturally result in lower RMS errors than predicting the ocean interior. To avoid this, we again normalize the RMS errors by a persistence forecast (that is a forecast of no change). These results show us the relative benefit of our CNN model over this simple baseline forecast. As discussed previously, normalizing the scores by a persistence baseline removes the impact of the difficulty of the forecasting problem, giving results which allow us to better compare the skill of the network in these different areas. (As described earlier, a normalized score of 1 implies the models are of equal skill, and scores lower (greater) than 1 show the CNN to have increased (decreased) skill over the persistence forecast.) We see here that normalized RMS scores for the ocean interior are an order of magnitude better than those at coastal points for all variables. Notably, performance next to land is worse than the simple persistence forecast when looking at temperature and Northward velocities, and when looking at the average over all variables combined. For SSH and Eastward velocity, near land the CNN gives only very marginal improvements over a persistence forecast. We see that while the network has notable skill predicting the ocean interior, near to land the network is no better than a simple persistence prediction.

4. Discussion

Our results show that the CNN does a great job of predicting the ocean evolution, with errors generally less than 5% of the standard deviation of the variable being modeled. However, performance differs between the physical variables, with temperature giving larger overall errors than other variables. When we focus on the impact of land, and look at performance near to land and in the ocean interior we see that the model performs much better in the open ocean than it does near land. One important finding of our results is that when assessing over the entire domain, issues at the boundary interface are often lost due to the orders of magnitude difference between the number of boundary points and interior ocean points. To assess if treatment of land is adequate, one needs to assess this in isolation from the ocean interior.

Regarding the poor performance near land, the representation of land used in this work provides the network with the information needed to theoretically capture the concept of land, and therefore of land–sea interactions. However, there are notable limitations of the approach used. Masking with zero values means land is treated in a similar way to the zero padding often used in CNNs. Liu et al. (2023) discussed the numerical artifacts which can occur as a consequence of this zero padding. We also note that by setting physical variables to zero at land points, we are choosing a value which has physical meaning outside of its use as land representation (points are set to zero prior to normalization). For example, considering

temperature, this means land–sea interactions are seen by the network as water of various temperatures at the ocean interior meeting (generally colder) 0° water within land. The standard effect of this representation elsewhere in the domain would be mixing and cooling of the ocean and generation of eddies due to the presence of sharp gradients. However, when ocean water meets land the resulting impact should be for there to be no change in temperature. With this representation, we are relying on the network having learnt, based entirely on the land–sea masks, which instances of 0° water next to warmer water should lead to mixing of temperatures, and which should lead to no mixing at all.

A more ideal approach would be to use a value which has no other physical meaning to mask variables. This would mean the network was less reliant on the land–sea masks to distinguish between which cases of 0° water next to other temperature water should result in mixing, and which are cases of ocean points next to land and should result in no mixing. In practice, however, this is tricky. Perceptually masking with “NaN” (Not a Number) values would capture what is meant by land points. However, as neural networks are based on matrix multiplication, any NaN values quickly propagate through the calculations and lead to NaNs throughout the domain. Another approach would be to use a value which is outside of the limits of the physical variables, for example, using –999 as the value for land points. However, this significantly changes the distribution of the data, and we found it results in poor performance (not shown). We feel that using zero, along with the land–sea masks, is the best available approach at present. Our results indicate that this implementation is sufficient for generally good predictions and to capture fundamental coastal dynamics in some regions. However, a better representation of land would likely improve performance, especially in areas where the dynamics are particularly complicated.

It should be noted that only one form of architecture was assessed within this work, and an alternative choice of architecture may improve performance near land. There are two potential issues associated with land. Firstly, the presence of land results in an irregular grid, which CNNs in their native form are not equipped to deal with. Liu et al. (2023) proposed a partial convolution method, intended for use with zero padding at image boundaries, which is used by Durand et al. (2024) to effectively deal with the irregularity of land in a CNN-based model. The results of the study by Durand et al. (2024) indicate that this approach would likely improve performance near land in an ocean model such as the one studied here. We also note that many recent applications of ML for atmospheric forecasts use graph neural networks (Lam et al., 2023) or transformers (Bi et al., 2023), both of which more naturally deal with irregular domains and may again show improvements for ocean models.

However, the second challenge is that the land itself impacts the ocean, impacting the dynamics directly. While the above methods solve the irregularity of the domain, they do so by removing the land entirely from the dataset. It is reasonable to expect that if spatial information (such as latitude and longitude) is included in the model inputs, any ML method could implicitly capture some of the impact of land. However, by excluding land entirely, we would remove the explicit dynamic impact from the dataset.

Another limitation of this work is that it considers just one configuration of a highly idealized bathymetry. Further work should assess performance in configurations with more realistic, irregular coastlines, and with varying bathymetry. Using a configuration with a variety of land boundary geometries within it, and with sloping bathymetry near land, may enable the network to better learn the dynamics of land–sea interactions and of the impact of bathymetry. Another potential avenue is to train the network on a number of different configurations, each with different coastline geometries. Again this approach may provide the network with a dataset which better demonstrates the underlying physics of interactions with land, enabling the network to better learn this.

An important point we highlight here is that the poor performance of our model near land may be a result of the *optimisation-use dichotomy*. There are many instances where the ways in which data-driven models are used does not strictly overlap with their optimization task. In this instance, we note that earth system models are often used to give localized information for numerous areas within the modeled region. By contrast, data-driven models are optimized to give good averaged performance. There may be times when these two aims overlap, but this is not necessarily the case. Instead, as seen here, there may be systematic biases within data-driven models, meaning specific areas and/or specific types of dynamics are

poorly modeled. While the optimization-use dichotomy exists to some extent in many modeling techniques, including process-based models, the fact that data-driven models are produced purely through an optimization procedure makes this issue more profound for these approaches. This potential mismatch between the ways in which data-driven models are optimized, and the ways in which they are used highlights the need for careful evaluation of data-driven models, and the importance of assessing many different components of their performance. It is of course possible to weight the loss function to avoid the issue seen here or to use attention mechanisms to focus the algorithm on coastal dynamics. However, in general, systematic biases may not be obvious a priori, and only notable after careful analysis. While we have only assessed a single architecture, here CNNs, and note there may be substantial improvements to performance near land with other architectures, this optimization-use dichotomy is not linked to any specific architecture, and may lead to issues with all data-driven approaches.

The issue of correctly modeling the impact of land within a ML-based model is certainly an area that would benefit from further consideration. This is an area where developing new ML techniques specifically suited to the geosciences, or adapting currently used techniques for the nuances of this application, could bring huge benefit.

We looked to the field of Computational Fluid Dynamics (CFD) to see what approach is taken here, as flow around a rigid body is a common CFD problem. A similar approach tends to be used, with physical variables set to zero over non-fluid regions, and an additional field given as input which identifies the location of fluid within the domain. There are, however, different approaches used for this location field (our land–sea masks). Guo et al. (2016) used a Signed Distance Function (SDF) which indicates not just the location of a boundary, but distance from it, along with whether a point is inside or outside of the rigid body. This gives more global information about the nearness of a location to the boundary. Later work in the study by Hennigh et al. (2017) instead used a simple binary input field (as is done in our work), and of particular interest here, they find that in their case the complex SDF of Guo et al. (2016) gives no notable improvement above their binary method. Ozaki and Aoyagi (2022) used a smooth profile function, which labels regions of fluid and non-fluid with 0 and 1, respectively, and the fluid–structure boundary is represented as numbers between 0 and 1. These papers all give good results in modeling a variety of CFD rigid body problems, using U-NET style CNNs as we have used here. However, very little analysis is given to the impact of the boundary treatment directly, with the exception of Ozaki and Aoyagi (2022), which briefly notes the model struggles at the boundary. It is worth considering if a more nuanced representation of land, in particular, a distance function similar to that used in the study by Guo et al. (2016) might provide benefit; however, without specifically analyzing results near the boundary, it is not possible to say if these methods give any benefit in the CFD cases studied. Added to this, modeling the ocean differs from a rigid body problem in that there are a number of coastlines, which have different impacts, and coastline shape and direction matter as much as distance from the coastline. It is not obvious that there are solutions from CFD research that would easily be applied in an ocean modeling application.

When looking at the poor performance for temperature predictions, this may be due to the nature of this data. Histograms of the data (Figures 12 and 13) show that temperature input data is far from Gaussian, with a complicated distribution. There is an absolute minimum, with a truncated distribution rather than a tail, and a number of peaks with varying intensity. By far, the largest peak is at the minimum, due to the large proportion of cold water in the domain. The global ocean is dominated by cold water at depth—the warmer surface waters make up a very small percentage of total water. As such, while the distribution here is specific to the configuration design, the large peak of very cold water, along with the lack of tail at the lower end of the distribution, is representative of the ocean in general. It is common practice in ML to standardize the dataset so that all variables exist on a similar scale, and with a balance of positive and negative values, as this can speed up learning. The distribution of temperature data is significantly different to other variables, especially the velocity variables, and so poses a challenge. Even when standardized the temperature data still has an extremely large peak right at the lower limit of the scaled data, rather than the majority of data being located near the center of the scale as is the case for the velocity data. The majority of temperature data becomes negative with

standardization (rather than the data being split between positive and negative values), and many of the data points remain of notably higher magnitude than the majority of other inputs, which may create an additional challenge to the network. We note that while SSH also has data peaks toward the end of the scales, this is much less extreme than the temperature data (note the scale on the y -axis in Figure 12), and we see in Figure 6 that SSH also shows poorer training scores compared to the velocity data. While all inputs impact the prediction of each variable, we would expect that the inputs for a specific variable more heavily influence prediction of that variable, that is, predictions of temperature are based on the current values of temperature, SSH, and both components of velocity, but we would expect them to be most heavily dependent on the current values of temperature. As such, it may be that because the temperature data is distributed with such an extreme peak at the very lower limit of the distribution, the network struggles more when predicting this.

5. Conclusion

Recent examples in the literature show there is tremendous potential for data-driven weather prediction systems, with results that are competitive with operational NWP forecast systems (Bi et al., 2023; Keisler, 2022; Pathak et al., 2022). However, the main focus to date has been on predicting atmospheric dynamics. Here, we focus on an ocean application, assessing the potential for ML to emulate GCMs of the ocean. Following Scher (2018), Scher and Messori (2019), Weyn et al. (2019) and Weyn et al. (2020), we develop a U-net style CNN architecture and use this to learn the dynamics of a channel configuration of MITgcm. A fundamental difference between modeling the ocean versus the atmosphere is the existence of land points which influence the flow but are not themselves modeled. We address the presence of land by providing the network with land–sea masks and masking all input and output data as zero at land points. While this approach works from a practical sense, there are conceptual issues with this, which may contribute to errors in the network predictions.

We show that the model performs well over the entire domain. Scatter plots show predictions are well matched to the true evolution of the GCM, and the RMS error over the test set is 2.69×10^{-3} , an order of magnitude better than a persistence forecast. While the persistence comparison is a fairly simple baseline, we feel it is sufficient for this “proof of concept configuration” and serves well to distinguish between the difficulty of the forecast problem and the skill of the neural network.

Spatial plots of RMS errors show that errors are largest in the areas we would expect where dynamics are most complicated, predominantly in the center of the channel, where there are strong temperature and SSH fronts. Errors are also high around land, though with some complexity to the pattern—generally performance is better on the Southern land boundary than the Northern one, and performance along the Northern boundary differs spatially across the variables. We see that along with exhibiting low errors, the predictions are qualitatively reasonable. In particular, resolution, intensity, and frequency of features in the predicted fields are comparable to those in the underlying GCM.

When we separate results near to land from those in the ocean interior, we see that performance is considerably worse near to land. Scatter plots for predictions of coastal points have a much broader spread than those for the ocean interior. The model tends to incorrectly predict zero values in a lot of cases and, in the case of temperature and Northward velocity, struggles to even capture the dominant signal. RMS errors normalized by a persistence forecast go from 1.38×10^{-1} in the ocean interior to 1.05 for coastal points, and results are roughly an order of magnitude worse for all variables. Near to land, the model adds no skill over a simple persistence forecast—though in the ocean interior, the model has considerable additional skill. These results indicate that the approach taken with land may be sufficient in some regions, but is serious lacking in others, and that a different approach is needed. These results also show that the performance over the entire domain is dominated by performance over the ocean interior, and near land points must be assessed on their own to give an indication of how well a model deals with land–sea interactions.

This work acts as a proof of concept for data-driven prediction systems for the ocean and shows these systems have promise. However, further development is needed. In particular, the representation of land

within CNN-based ocean models needs careful consideration and assessment. This performance may be improved with alternative ML-based approaches, such as graph neural networks or transformers, or by using partial convolutions within the CNN approach. There may also be considerable scope for ML methods to be created or adapted, in order to improve representation of land within ML models of the ocean. Any such methods must be assessed by analyzing results near to land rather than over an entire ocean domain.

Open peer review. To view the open peer review materials for this article, please visit <http://doi.org/10.1017/eds.2024.49>.

Acknowledgements. Many thanks to the two anonymous reviewers who gave feedback on this work for their insightful comments and suggestions.

Author contribution. Conceptualization: RF. Formal analysis: RF. Investigation: RF. Methodology: RF. Software: RF. Writing—original draft: RF. Writing—review and editing: RF, PH, DCJ, DM, BP, and ES. Supervision: PH, DCJ, DM, BP, and ES.

Competing interest. The authors declare no competing interests.

Data availability statement. Code used for this work can be found at <https://zenodo.org/records/10817911>. Due to its size, the data is available on request from the authors.

Ethics statement. The research meets all ethical guidelines, including adherence to the legal requirements of the study country.

Funding statement. RF was supported by the UK Natural Environment Research Council [grant number NE/L002507/1].

References

- Arcomano T, Szunyogh I, Pathak J, Wikner A, Hunt BR and Ott E (2020) A machine learning-based global atmospheric forecast model. *Geophysical Research Letters* 47.
- Bi K, Xie L, Zhang H, Chen X, Gu X and Tian Q (2023) Accurate medium-range global weather forecasting with 3d neural networks. *Nature* 619,533–538.
- Chen K Han T, Gong J, Bai L, Ling F, Luo J-J, Chen X, Ma L, Zhang T, Su R, Ci Y, Li B, Yang X and Ouyang W (2023a) Fengwu: pushing the skillful global medium-range weather forecast beyond 10 days lead. *arXiv*.
- Chen L, Zhong X, Zhang F, Cheng Y, Xu Y, Qi Y and Li H (2023b) Fuxi: a cascade machine learning forecasting system for 15-day global weather forecast. *npj Climate and Atmospheric Science* 6.
- Clare MC, Jamil O and Morcrette CJ (2021) Combining distribution-based neural networks to predict weather forecast probabilities. *Quarterly Journal of the Royal Meteorological Society* 147, 4337–4357.
- Dueben PD and Bauer P (2018) Challenges and design choices for global weather and climate models based on machine learning. *Geoscientific Model Development* 11, 3999–4009.
- Durand C, Finn TS, Farchi A, Bocquet M, Boutin G and Ólason E (2024) Data-driven surrogate modeling of high-resolution sea-ice thickness in the arctic. *Cryosphere* 18, 1791–1815.
- Furner R, Haynes P, Munday D, Paige B Jones DC and Shuckburgh E (2022) A sensitivity analysis of a regression model of ocean temperature. *Environmental Data Science* 1, 11–12.
- Guo X, Li W and Iorio F (2016) Convolutional neural networks for steady flow approximation. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 13–17 August, pp. 481–490.
- Hennigh O (2017). *Automated Design using Neural Networks and Gradient Descent*. *ArXiv*, abs/1710.10352.
- Keisler R (2022) Forecasting global weather with graph neural networks. *arXiv*.
- Lam R, Sanchez-Gonzalez A, Willson M, Wirsberger P, Fortunato M, Alet F, Ravuri S, Ewalds T, Eaton-Rosen Z, Hu W, Meroze A, Hoyer S, Holland G, Vinyals O, Stott J, Pritzel A, Mohamed S and Battaglia P (2023) Learning skillful medium-range global weather forecasting. *Science* 382, 1416–1421.
- Liu G, Shih KJ, Wang T-C, Reda FA, Sapra K, Yu Z, Tao A and Catanzaro B (2023) Partial convolution based padding. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 6096–6110.
- Marshall J, Adcroft A, Campin JM, Hill C and White A (2004) Atmosphere-ocean modeling exploiting fluid isomorphisms. *Monthly Weather Review* 132, 2882–2894.
- Munday DR, Johnson HL and Marshall DP (2015) The role of ocean gateways in the dynamics and sensitivity to wind stress of the early Antarctic circumpolar current. *Paleoceanography* 30, 284–302.
- Nguyen T, Brandstetter J, Kapoor A, Gupta JK and Grover A (2023) Climax: a foundation model for weather and climate. *Proceedings of Machine Learning Research* 202, 25904–25938.
- Ozaki H and Aoyagi T (2022) Prediction of steady flows passing fixed cylinders using deep learning. *Scientific Reports* 12, 447.
- Pathak J, Subramanian S, Harrington P, Raja S, Chattopadhyay A, Mardani M, Kurth T, Hall D, Li Z, Azizzadenesheli K, Hassanzadeh P, Kashinath K and Anandkumar A (2022) Fourcastnet: a global data-driven high-resolution weather model using adaptive Fourier neural operators. *arXiv*.

- Price I, Sanchez-Gonzalez A, Alet F, Ewalds T, El-Kadi A, Stott J, Mohamed S, Battaglia P, Lam R and Willson M (2023) Gencast: diffusion-based ensemble forecasting for medium-range weather. *arXiv*, pp. 2023–2035.
- Rasp S and Thuerey N (2021) Data-driven medium-range weather prediction with a resnet pretrained on climate simulations: a new model for weatherbench. *Journal of Advances in Modeling Earth Systems*. 13, e2020MS00240.
- Ronneberger O, Fischer P and Brox T (2015) U-net: convolutional networks for biomedical image segmentation. In *Lecture Notes in Computer Science (Including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 9351, pp. 234–241. Cham: Springer.
- Scher S (2018) Toward data-driven weather and climate forecasting: Approximating a simple general circulation model with deep learning. *Geophysical Research Letters* 45, 12,616–12,622.
- Scher S and Messori G (2019) Weather and climate forecasting with neural networks: Using general circulation models (GCMs) with different complexity as a study ground. *Geoscientific Model Development* 12, 2797–2809.
- 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 and Song J (2024) Xihe: a data-driven model for global ocean eddy-resolving forecasting. *arXiv*.
- Watson-Parris D, Rao Y, Olivie D, Seland N, Camps-Valls P, Stier G, Bouabid P, Dewey S, Fons M, Gonzalez E, Harder J, Jeggle P, Lenhardt K, Manshausen J, Novitasari P, Ricard ML and Roesch C (2022) Climatebench v1.0: a benchmark for data-driven climate projections. *Journal of Advances in Modeling Earth Systems* 14.
- Weyn JA, Durran DR and Caruana R (2019) Can machines learn to predict weather? using deep learning to predict gridded 500-hpa geopotential height from historical weather data. *Journal of Advances in Modeling Earth Systems* 11, 2680–2693.
- Weyn JA, Durran DR and Caruana, R. (2020). Improving data-driven global weather prediction using deep convolutional neural networks on a cubed sphere. *Journal of Advances in Modeling Earth Systems* 12.
- Xiong W, Xiang Y, Wu H, Zhou S, Sun Y, Ma M and Huang X (2023) AI-GOMS: large AI-driven global ocean modeling system. *arXiv*.

Appendix A. Histograms of MITgcm dataset

Histograms of input and target variables are shown in Figures A1 and A2. We can see that while target data generally follows a Gaussian distribution, this is not the case for input data. In particular, we see that the temperature, SSH, and wind forcing data all exhibit multiple peaks, with temperature also lacking any tail at the lower end of the distribution. Note we have used fewer bins for the wind forcing histogram—there is no temporal variability in this data, and this combined with the spatial discretization means the data only exists over a limited number of bins in comparison to other variables. In temperature, surface temperature flux, and both components of velocity, we see that the tails (where they exist) are very long—note the range of the plots represents the range of data with tails so small that they are not visible in most cases. We see the distribution of the Eastward component of velocity is skewed toward positive values, inline with the dominant zonal flow in the configuration.

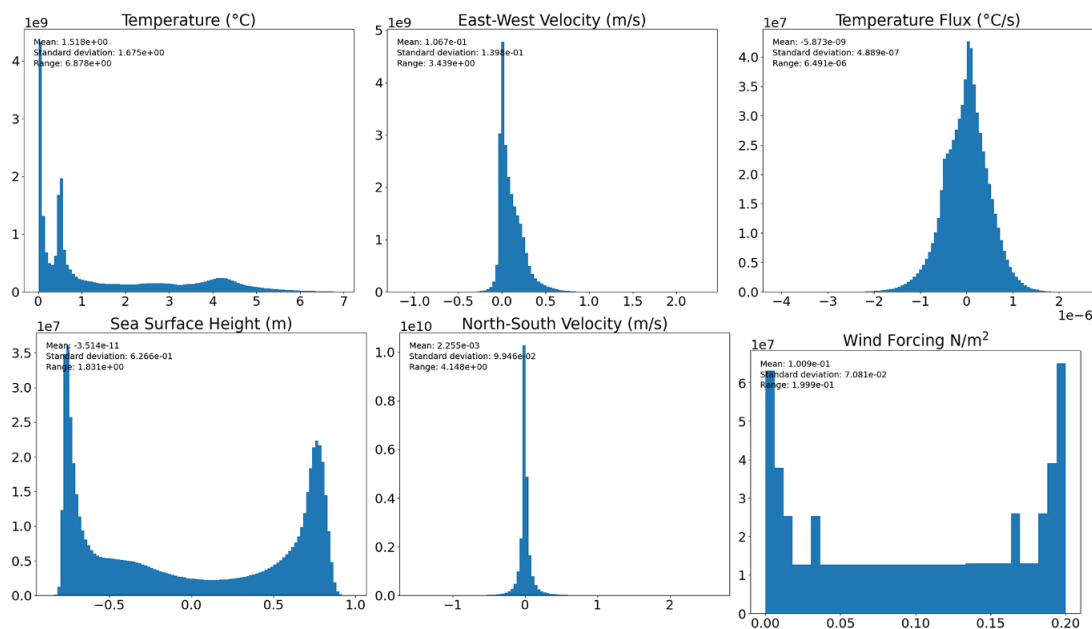


Figure A1. Histograms of input variables.

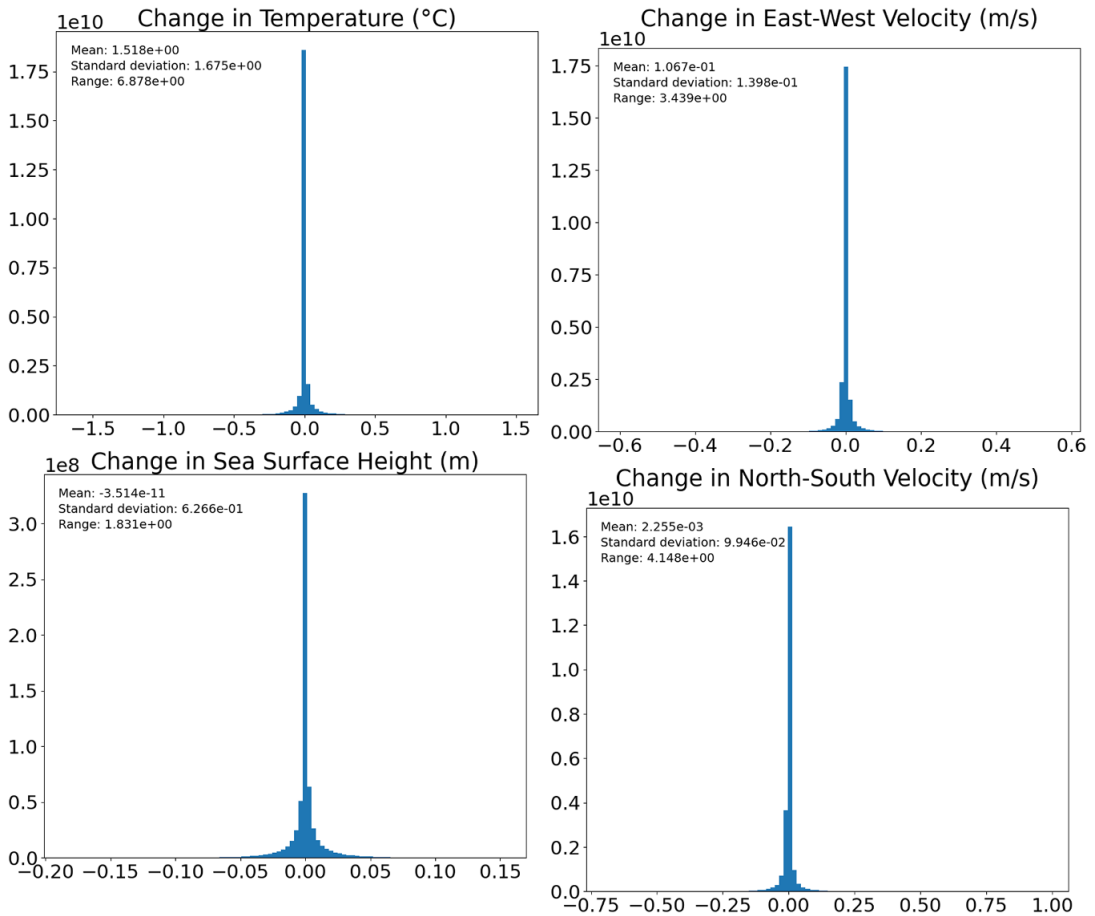


Figure A2. Histograms of target variables—that is, the change in each variable over 12 hours.

The target data follows a more Gaussian distribution with all variables exhibiting a single peak. Again the distributions show very long, small tails. Here, we also see that the peaks are very large, with the vast majority of data situated around the zero mark.

The large, varying ranges and small tails mean that normalizing the data using the standard deviation results in data which still have notably different ranges for the different variables. For this reason, we normalize by subtracting the mean and dividing by the range. This has the desired impact of ensuring all data is transformed to exist on roughly similar scales for all variables.