Enhancing the accuracy of metocean hindcasts using machine learning models

Mariana O. Costa[1], Ricardo M. Campos[2,3], C. Guedes Soares[1*]

[1] Centre for Marine Technology and Ocean Engineering (CENTEC), University of Lisbon, Instituto Superior Técnico, Av. Rovisco Pais, 1049-001 Lisboa, Portugal.

[2] Cooperative Institute for Marine and Atmospheric Studies (CIMAS), University of Miami, 4600 Rickenbacker Causeway, Miami, FL 33149, USA.

[3] NOAA/Atlantic Oceanographic and Meteorological Laboratory (AOML), 4301 Rickenbacker Causeway, Miami, FL 33149, USA.

*Corresponding Author e-mail address: c.guedes.soares@centec.tecnico.ulisboa.pt

Abstract

To improve the quality of wind and wave forecasts post-processing algorithms are developed to bias-correct the variables of interest using machine learning models trained with buoy data. Two types of deep neural networks are developed and studied to improve the accuracy of ERA5 hourly data of 10-m wind speed (wspd) and significant wave height (swh) at two locations with distinct metocean conditions. The algorithms are the multilayer perceptron (MLP) and the long short-term memory (LSTM) neural networks. The latter is built based on a range of previous time steps included in the input, evaluating the optimal value. Several metrics such as bias, scatter index, root mean square error and correlation coefficient are used to evaluate model performances. After extensive examination and experimentation regarding feature selection, filtering windows, and model architecture, the results show better improvements for swh than wspd, and that LSTM outperforms MLP significantly. The original correlation coefficients between ERA5 and observations for swh have been improved and the ERA5 RMSE has been reduced with the LSTM post-processing model.

Keywords: neural networks, long short-term memory, wind and wave modeling, reanalysis, feature selection, bias correction.

1. Introduction

Accurate wind and wave information is fundamental for many activities, including coastal safety, marine transportation and alerts of extreme events. In terms of in-situ observations, buoys are considered to provide the most precise measurements and present a pivotal source of

information (e.g., Menéndez et al. 2008, Gemmrich et al. 2011); however, the data is limited to discrete locations, normally close to coasts, and they often have various gaps or limited duration. The progress in numerical modelling has notably supplemented the various sources of wind and wave data with an improved spatial and temporal resolution, through the reanalysis projects. Reanalyses incorporate observations from a wide range of platforms while implementing the same numerical model and data assimilation technique for consistency – becoming an important and reliable source of metocean data for ocean engineering applications.

Data assimilation in numerical weather prediction models has had an important role in improving the accuracy of the predictions. In general, two main approaches have been applied in data assimilation, the sequential methods and the variational approaches. Both methods start from a background field that has been predicted by the model, which is compared with observations to determine the error, which becomes the correction to be imposed on the predicted values to improve them.

In sequential schemes, the background condition at the forecast time step is corrected by the observations at that time instant and is used as initial conditions for the next forecast step. The variational methods recognise that the errors in predictions that occur before and after the time instant under consideration are important to improve the background field and thus they use observations more spread in time than sequential methods. Thus, variational methods are more relevant for reanalyses, while sequential methods are generally used in forecast situations.

Representative sequential schemes are the optimal interpolation (Lionello et al. 1992; Rusu and Guedes Soares, 2015), and the successive interpolation method (Breivik and Reistad, 1994; Rusu and Guedes Soares, 2014), which have been applied in global and regional studies.

There are approaches based on the Kalman filter, which also use present and past information for data assimilation (Voorrips et al. 1999). They have been extended to different forms, including the ensemble Kalman Filter (Evensen 2013). They have been applied as a postprocessing approach to the forecasted significant wave height (Galanis et al. 2009) or to the significant wave height and mean period (Almeida et al. 2016).

James at al. (2018) and Feng (2020) have used machine learning models of multilayer feedforward perceptron (MLP) with supervised training as surrogate models for predictions of SWAN assuming steady-state conditions. This allowed very quick predictions of surf conditions based on offshore wave forecasts.

The use of NNs for environmental analysis and forecasts has promptly escalated over recent years. Artificial neural network (ANN) models are being applied to significant wave height prediction. For example, Deo and Sridhar Naidu (1998) proposed a feedforward network to predict sea wave height in real-time. Compared with the autoregressive models (e.g. Guedes Soares and Cunha 2000), their method demonstrated a more general, flexible, and adaptable capability. While Krasnopolsky and Schiller (2003) used forward and inverse NN models in remote sensing, this same general approach was used by Wahle et al. (2015) to assimilate data into WAM model predictions. They used ANNs to emulate the wave model and its inverse by correcting the wind fields in one step and the initial conditions in the next step. Combinations of the used ensemble Kalman filter together with an ANN wind-wave model can also be found (Zaman et al. 2010).

Londhe et al (2016) have used ANNs to model the error between wave forecasts and buoy measurements in four locations and the used the model to correct forecasts in real time, improving thus the forecasts at the buoy locations. Four ANN models were proposed by Elbisy

and Elbisy (2021) namely, the multilayer perceptron neural network (MPNN), cascade correlation neural network (CCNN), radial basis function neural network (RBFNN), and general regression neural network (GRNN). The overall performance of the RBFNN was found to be more accurate than those of the other models, and the CCNN model exhibits the worst predictive capabilities. A multilayer feed-forward neural network with the error backpropagation algorithm was adopted by Zheng et al. (2022). Oo and Zhang (2022) have used a ANN to model the error of predictions by MIKE 21 and then was coupled back with the model to improve its prediction skill.

Mandal and Prabaharan (2006) used a recurrent neural network (RNN) for significant wave height prediction and found that the correlation coefficient of the RNN output is higher than that of the feedforward network. Pirhooshyaran and Snyder, (2020) have used RNN together with other techniques to model wave forecasts by WW3 and extrapolate them.

One of the time-recurrent neural networks, proposed by Hochreiter and Schmidhuber (1997), is the Long Short-Term Memory (LSTM) network, which can choose to remember or forget long-term information through a forget gate. Even though LSTMs are well suited for time series analysis the use of LSTM was introduced by Fan et al., (2020), to predict the significant wave height at several locations and its results are compared with those of ANN, being one of the first papers to conduct such study. Other studies that used LSTM trained with buoy data are Ni and Ma (2020)

Wei (2021) also used buoy data to forecast significant wave height various hours ahead and discussed how to choose the training set for LSTM modelling. Ang et al (2022) have also used LSTM trained by long term hindcast data produced by WW3.

LSTM models have also been used to model time series of wave heights instead of longterm series of significant wave height. Kagemoto (2020) used LSTM to forecast wave trains, generating reasonably accurate forecasts, while Ma et al (2022) addressed the prediction of short-crested wave fields.

Convolutional LSTM networks trained with historical sea surface wind and wave have been applied to data from South and East China Seas by Zhou et al. (2021) and Bai et al (2022) and to a dataset of North West Pacific by Gao et al (2023). Other applications of convolutional neural networks (CNN) were made by Jing et al (2022) and Huang et al (2022).

Although NNs became very popular, most of the forecast studies have been aimed at applying NN models directly to predict wave heights and surface winds as target variables. Campos et al. (2019) proposed an alternative methodology using a hybrid model, joining the numerical wave model with NNs. The numerical model predicts the variables of interest while the target of the NN is to predict the residue (i.e., the difference between the measurement and the model), which is recombined to provide an accurate estimation of wave heights and surface winds. The experiment was concentrated on the Brazilian coast and delivered successful results. Still, only forecasts have been evaluated in such a framework, and the present study is an extension of this methodology by applying it to hindcasts.

By optimally combining observations and models, reanalysis is an excellent substitute for when measurements are not available, supporting consistent “maps without gaps”. The metocean data from the European Centre for Medium-Range Weather Forecasts (ECMWF) ERA reanalyses, evaluated and discussed by Stopa and Cheung (2014) and Campos and Guedes Soares (2016), is an example of a publicly available and widely used global dataset.

The ECMWF ERA5 reanalysis, which has upgraded performance compared to its corresponding predecessor ERA-I reanalysis, represents an important data source for the ocean modelling community (Hersbach et al., 2020). The constantly irregular coverage and resolution of observations used in the assimilation may still represent a source of non-physical variations that require attention in the modelling of ocean wave climate, as well as the increased uncertainties under extreme conditions (Stopa and Cheung, 2014). The shortcomings still present in ERA5 estimations, specifically for significant wave height and 10-m wind speed, are evaluated and adjusted in this study by incorporating deep neural networks (DNNs) in the estimation process trained with National Data Buoy Center (NDBC) buoys data.

Given the above, the present paper is focused on developing hybrid models composed of ERA5 reanalysis estimations and DNN models, in an attempt to improve the hindcast estimates even further. The main goal is to reduce both the systematic and scatter errors of significant wave height and surface winds estimates in the North Atlantic Ocean by developing different NN models. The structure of this study starts with the description and analysis of the data in section 2, to create the best feature space to train the models that are introduced in section 3, also including all the experiments and optimal structures, culminating in the final results, discussed using four error metrics. Section 4 presents the conclusions, challenges and suggestions for the next steps.

2. Data description

The observations used for the present study consist of global quality-controlled buoy measurements from the NOAA's National Data Buoy Center (NDBC). Certain events can make buoys fail, such as storms (Rao and Mandal, 2005), maintenance periods, and navigation accidents, among others., leading to data gaps and thus discontinuities in the buoy's time series,

lasting from the causing event until the buoy is repaired. Selecting a relatively large period with the least amount of gaps and missing data is a fundamental step since one of the algorithms of interest, LSTM, retains useful information about previous data in the sequence to help with the processing of new data.

After quality control, the selected period corresponds to three years from 2014 to 2016, from NDBC buoys 41040 (14.540 N 53.329 W) and 41048 (31.831 N 69.573 W) in the North Atlantic Ocean with water depths of 5159 and 5394 meters, respectively. The exact locations can be seen in Figure 1. The selection of these two buoys derives not only from the lowest number of gaps among stations but also the fact that metocean conditions are distinct, which enables better generalization of the conclusions.

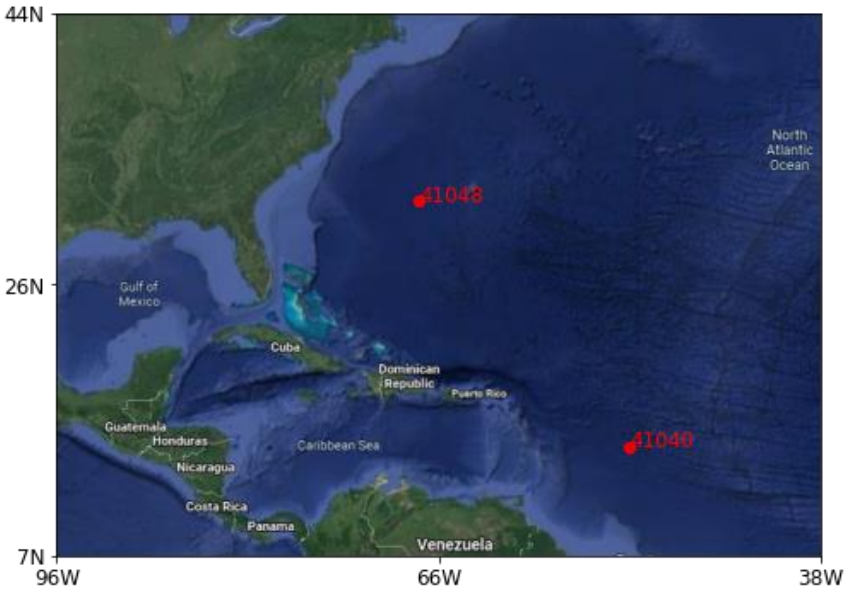

Figure 1 - Location of the two NDBC buoys selected for this study.

The initial data set contains information about 36 variables estimated by ERA5, which uses a temporal resolution of 3 hours and a spatial resolution of 31km. A few adjustments are made to generate the input data space by applying the following variable transformations:

$$sint = sin\left(\frac{2\pi}{365}time\right), \qquad cost = cos\left(\frac{2\pi}{365}time\right) \tag{1}$$

$$ud = sin\left(\frac{2\pi}{180}d\right), \qquad vd = cos\left(\frac{2\pi}{180}d\right) \tag{2}$$

$$wspd = \sqrt{u^2 + v^2} \tag{3}$$

The sine and cosine of time, introduced in Eq. 1, allow the inclusion of annual cycles and seasonal effects in the mapping. The directional variables are replaced by their corresponding sine and cosine as well, introduced in Eq. 2, where d is the direction. Hence, the direction angles can be properly used by the machine learning model, avoiding errors for example when a direction of -180 comes after 179 or 0 after 360. The wind components u and v at 10 meters are converted to wind speed, according to Eq. 3. The log function is applied to significant wave height following the work of Campos et al. (2019), where it is stated that it has been confirmed that such transformation boosts results considering the more homogeneous distribution of values.

The final dataset is composed of 43 features containing information about the two buoy locations for the period 2014-2016. The buoy measurements of swh and wspd are subtracted from the corresponding ERA5 estimates, yielding the estimated residues, which are the target variables of the NN models. The definition (Def.) of each feature and corresponding abbreviation (Abb.) are presented in Table 1. Before feeding data into the NNs, the features must be analysed, selecting the optimal set with minimum complexity, i.e., transforming data from a high-dimensional space into a low-dimensional space without losing the important information. Such analysis is handled in the following subsection. The first two years of data, 2014 and 2015, are used for training and the whole year of 2016 is used as a test set.

As previously mentioned, the two buoy locations are chosen not only due to their lower number of gaps in the time series but also because of the distinct ocean conditions, more

specifically, of the variables of interest, swh and wspd. Such contrast can be visualized in Figure 2, where the time series of the two buoys are overlapped for each variable. The time-series variability is much higher in buoy 41048, as well as the maximum peak values.

Table 1 - Definition (Def.) and abbreviation (Abb.) of the study variables, composed by 43 features and 2 targets.

Def.	Abb.	Def.	Abb.
Significant wave height of first swell partition	swh1	Mean wave period of first swell partition	mwp1
Significant wave height of second swell partition	swh2	Mean wave period of second swell partition	mwp2
Significant wave height of third swell partition	swh3	Mean wave period of third swell partition	mwp3
U-Component of wind	u850	V-Component of wind	v850
Mean sea level pressure	msl	Air temperature	atmp
2 meter dewpoint temperature	dewp	Period corresponding to maximum individual wave height	tmax
Maximum individual wave height	hmax	Mean 0 crossing wave period	m0wp
Wave spectral directional width	wdw	Wave spectral directional width for wind waves	dwww
Wave spectral directional width for swell	dwps	Peak period	pp1d
Mean wave period	mwp	Significant height of wind waves	shww
Mean period of wind waves	mpww	Significant height of total swell	shts
Mean period of total swell	mpts	Instantaneous 10 metre wind gust	i10fg
Sea surface temperature	sst	10 meter wind gust since previous post-processing	10fg
Geopotential height	hgt	Julian days cosine	cost
Julian days sine	sint	U-Component of mean wave direction	umwd
V-Component of mean wave direction	vmwd	U-Component of mean wave direction of first swell partition	umwd1
V-Component of mean wave direction of first swell partition	vmwd1	U-Component of mean wave direction of second swell partition	umwd2
V-Component of mean wave direction of second swell partition	vmwd2	U-Component of mean wave direction of third swell partition	umwd3
V-Component of mean wave direction of third swell partition	vmwd3	U-Component of mean direction of wind waves	umdww
V-Component of mean direction of wind waves	vmdww	U-Component of mean direction of total swell	umdts
V-Component of mean direction of total swell	vmdts	Logarithm of significant wave height	Lswh
Wind speed	wspd		
Residue of swh (target)	**e_swh**	**Residue of wspd (target)**	**e_wsp**

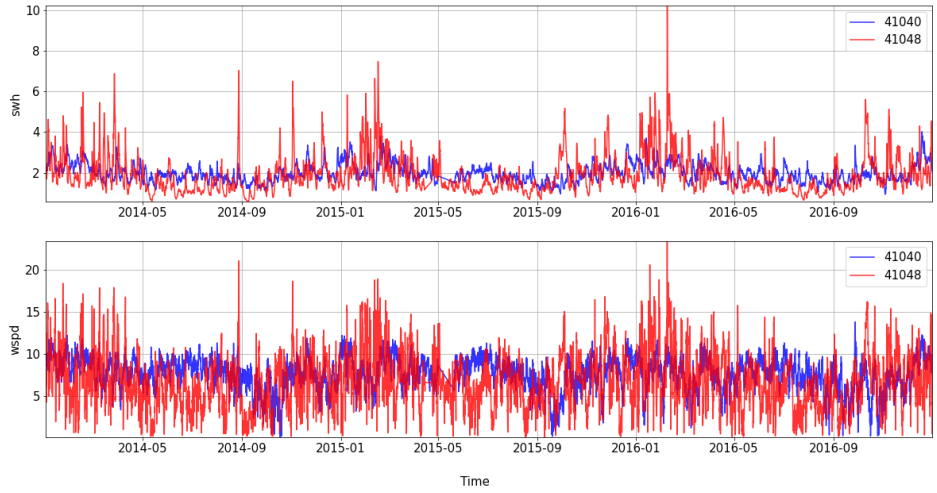

Figure 2 - Time series of significant wave height (top) and wind speed (bottom) for the period 2014-2016. The plots in blue represent the measurements of buoy 41040 and the plots in red represent the measurements of buoy 41048.

As the estimated residues are the core of this study, a total of four metrics are introduced to deepen the understanding of the data. The selection is based on the study of Mentaschi et al. (2013) and following the assessment works in Campos et al. (2019, 2020). The error metrics are:

$$NBias = \frac{\sum_{i=1}^{n}(x_i - y_i)}{\sum_{i=1}^{n} y_i} \tag{4}$$

$$SI = \sqrt{\frac{\sum_{i=1}^{n}[(x_i - \bar{x}) - (y_i - \bar{y})]^2}{\sum_{i=1}^{n} {y_i}^2}} \tag{5}$$

$$NRMSE = \sqrt{\frac{\sum_{i=1}^{n}(x_i - y_i)^2}{\sum_{i=1}^{n} {y_i}^2}} \tag{6}$$

$$CC = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(x_i - \bar{x})}{\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2 \sum_{i=1}^{n}(x_i - \bar{x})^2}} \tag{7}$$

The normalized bias (NBias), measures the systematic error; the scatter index (SI) measures the scatter error; the normalized root mean square error (NRMSE) combines the systematic and

scatter components, and the correlation coefficient (CC) evaluates the signal trends comparing model and observation. Therefore, Eqs. 4 to 7 describe the metrics selected, where x is the model data, y is the measurement, and the overbar indicated the arithmetic mean. The normalized non-dimensional metrics (NBias, SI and NRMSE) can be interpreted as ratios, or percentage errors divided by 100.

Note that ERA5 estimates are generally accurate on average, as shown in section 3. Thus, boosting estimates is a challenge that, if accomplished, can recreate a time series of winds and waves with high accuracy, which are especially important when measurements are unavailable.

2.1 Feature selection

One of the main steps in machine learning is feature selection and extraction, two ways of dimensionality reduction. The common weakness of machine learning algorithms is the so-called dimensionality curse, which makes models inefficient or even useless when solving large-scale problems, as explained in Domingos (2012). Hence, simplifying, achieving scalability, low computational complexity, and efficient performance have become a priority in modelling (Langley et al., 1994; Blum and Langley, 1997).

Feature selection is the process of discarding irrelevant and redundant data (Zhao et al., 2010), thus obtaining a representative subset that, ideally, gathers an identical amount of information as the original dataset and enhances learning efficiency. Most of the time, raw data has many noninformative features, and poor-quality input produces poor-quality output. Different feature selection strategies broadly fall into three categories: filter, wrapper, and embedded algorithms. Filter models rely on the general characteristics of data, evaluating features without involving any learning algorithm. Wrapper requires a predetermined learning algorithm and uses its performance as an evaluation criterion to select features. Algorithms with

embedded models perform feature selection and training of the model in parallel. The pros and cons of these models can be summarized as follows.

Filter methods are faster, and independent of the learning model while avoiding overfitting. Independence is an advantage since the choice of the learning model is not restricted but comes with the disadvantage of not interacting with it. Wrapper methods are simple, they interact with the learning model, and usually give the best performances, although they are prone to overfitting and are computationally intensive. Embedded methods lie in between the other two regarding performance and computational complexity. An advantage over filters is the interaction with the learning model, and an advantage over wrappers is the complexity and being less prone to overfitting (Li et al., 2017).

Feature extraction is the process of combining the original variables into new features, reducing the sources required to describe a large set of data without losing important information. Principal Components Analysis (PCA; Holland, 2008), is one of the most recognized statistical tools for feature extraction, defined as an orthogonal linear transformation. It is sensitive to scaling, and in multivariate cases, implementation requires data to be standardized so that all variables have a mean of 0 and a standard deviation of 1. This multivariate technique analyses a dataset with the goal of extracting the important information, representing it as a set of new orthogonal variables called principal components, each explaining the maximum amount of data's variance in decreasing order. In other words, the first principal component is the direction that maximizes the variance of the projected data, the second greatest variance lies in the second principal component, and so on. The transformed space has the same dimension as the original space. However, some of the higher order principal components are expected to be discarded, depending on the percentage of total variance intended to be kept.

Common thresholds for dimensionality reduction are 90%, 95% or even 99%. The purpose of dimensionality reduction is not only the decreased complexity in processing high-dimensional datasets and the removal of non-informative data but also when variables are noisy. PCA can have the effect of concentrating much of the signals into the first principal components, while the latter principal components may be dictated by noise, and thus can be discarded without much loss (Sophian et al., 2003; Skittides and Früh, 2014; Segreto et al., 2014).

In this research, both feature and time spaces are highly dimensional, thus four feature selection methods are investigated. The correlation and multicollinearity between variables is the first step to feature selection (filter method). Multicollinearity is a very strong linear correlation between two or more features, which leads to unstable models of excessive complexity. Variance inflation factor (VIF), (Katrutsa and Strijov 2017), is widely applied to detect such relationships. For each j-th feature, VIFj is calculated as $1/(1 - Rj2)$, where Rj2 is the coefficient of determination, and ranges from 1 upwards (1 if orthogonal). There is no formal VIF value for determining the presence of multicollinearity, although Paul (2006) stated that practical experience demonstrates that a value exceeding 5 or 10, is indicative of high multicollinearity between the present independent variable and the others. After setting the desired threshold, variables should be dropped iteratively, starting with the one with the greatest VIF value until all are below that threshold. In this study, it was decided to adopt the more conservative threshold, equal to 5.

A slight adaptation of the VIF method is designed here, where Lswh, wspd, cost, and sint are specified in the algorithm not to be eliminated during the dropout process; the first two because they are known to be fundamental in the prediction of the target variables, while the latter is intentionally added to the feature space so that the learning process has access to the time of the

year. Therefore, even if one of these variables has the maximum score at some step, which happens, it would be ignored and the next largest value would be chosen. After applying VIF, 15/16 out of the 43 variables are eliminated in buoy location 41040/41048, having then a reduced variable space, although still with cardinality 28/27. The eliminated variables due to multicollinearity are tmax, hmax, mwp, 10fg, shww, msl, shts, mpww, i10fg, umdts, m0wp, swh1, sst, vmwd and mpts in buoy 41040, and tmax, hgt, hmax, i10fg, shts, mwp, shww, sst, 10fg, atmp, mpww, m0wp, mpts, umwd, vmwd and swh1 in buoy 41048 (description of each variable in Table 1). At the end of the procedure, the variables that are specified not to be eliminated still have acceptable VIF values, with a few exceeding the more conservative threshold yet still inferior to the threshold most research papers consider, a value of 10. The maximum value is for sint in buoy 41040, with VIF equal to 7.37.

In Ebrahimi-Khusfi et al. (2021), a combination of different modelling and feature selection methods was adopted to identify the best approach for predicting the number of dusty days around the desert wetlands. It was observed that by only removing the problematic collinearity effects without performing any further feature selection, most models employed could successfully predict this climatic variable with high accuracy. The work in Jörges et al. (2021) concerns the prediction of nearshore significant wave height (swh) based on LSTM neural networks, a similar environment to the one presented in this paper. The method chosen for feature selection was based on the Pearson correlation coefficient (r) between features and target, where features with $r>0.3$ to swh were selected. However, this selection does not consider non-linear influences and in the present environment, none of every single feature is significantly correlated to the target variables. Thus, such an approach is not included here.

As the second step to feature selection, a wrapper method is thought to be a convenient approach. Initially, the popular Recursive Feature Elimination (RFE; Guyon et al., 2002) was applied due to its easy configurable nature and robust performance. It is a type of backward selection method that works on a feature ranking system. The main shortcoming is the problem of computational time consumption, verified in exploratory assessments. RFE gives a final ranking for the attributes and the most suitable features are selected based on a threshold. Such an approach led to a reduced dataset yet still with high cardinality. So, another approach is considered to drastically reduce the input space, Sequential Forward Selection (SFS). Instead of having all the features and recursively eliminating one by one, this algorithm starts with a null model and fits the model with each feature one at a time. The process is repeated until the desired number of features is included.

Random Forest Regressor (RF) algorithm (Callens et al. 2020) was the chosen base model, a meta estimator that fits several classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. Another advantage of this choice is the fact that RFs can handle nonlinear parameters efficiently (Campos et al. 2021). To significantly limit the input space, the threshold is set to six features only. A RF with 50 decision trees and a minimum of 10 samples required to be at a leaf node, to reduce complexity, is fit to SFS and the whole process is executed 10 times to account for the stochastic nature of the algorithm. The resulting optimal features passed through some additional adjustments as it involves a stochastic element and some variables are fundamental, for instance, Lswh and wspd. This adjustment also follows the suggestions of previous studies. Therefore, additional testing is conducted and ultimately the final input sets are listed in Table 2. In buoy

41040 the dimension of the input is even inferior to the established threshold as it delivered even better results.

Table 2 - Optimal feature space after two feature selection steps.

Buoy	Features
41040	Lswh, v850, vmwd3, wspd
41048	Lswh, u850, dwww, vmwd2, vmdww, wspd

It should be noted that since the ranking criterion of wrappers is computed with information about a single feature, there is no effect on correlation. Thus, having these two feature selection steps is favourable.

After this selection of features and removal of redundancies, PCA is included in the process of feature selection/reduction and employed as a third and last step of dimensionality reduction to compress a bit more and possibly filter noise. Yet this is only employed in buoy 41048, reducing the feature space to five components, as with only four features in buoy 41040, 95% of the total variance includes four components, not reducing the feature space any further.

3. Deep Learning Models for Multi-Output Regression and Evaluation

Neural networks (NNs) are being broadly utilized in multiple complex and challenging subjects in engineering and science, as they have a remarkable ability to model the non-linear relationships between inputs and outputs (Gurney, 2018). Deep learning models are built with deeper architectures in neural networks. The term “deep” refers to the use of multiple layers in the network. Two types of NNs are chosen for this study, Multilayer Perceptron (MLP) and Long Short-Term Memory (LSTM), formally described in the following respective subsections.

There are challenges to face when training NNs. Apart from data selection and proper scaling, one of the difficulties is a large number of hyperparameters one has to deal with when

designing the architecture of the model. These could be the learning rate, the activation function, weight of regularization technique, optimizer, weight initialization, etc. There is no universal "best choice" that is consistently good across different problem types, network architectures, activation functions, and data sets, as optimization is problem-dependent. Neural networks can be very sensitive to the initialization strategy that is used. Therefore, it is important to find ways to better organize the hyperparameter tuning process, as proper tuning requires many experiments and runs.

Python 3 is the software of choice for all experiments, specifically the keras library for the deep learning models. To address the hyperparameter problem, it was decided to use Talos, a python package for solving complex neural network models and determining the right combination of the parameters. The method works as an exhaustive search approach with a chosen parameter grid. There is the option to downsample the grid search by running only a fraction of the total combinations, which is very useful when the parameter space is too big. The best model can be found by running the code one time, instead of running the code after each change of a single parameter - saving time and making it easier to find the best combinations with the lowest loss values.

3.1 Multilayer Perceptron (MLP)

Multilayer Perceptron (MLP; Rumelhart et al., 1985), is considered the most common and most widely used type of NNs that can properly model complex relations (Hornik et al., 1989). Three (or more) types of fully connected layers constitute its structure: the input layer, where the model inputs are initiated; the output layer, where the results of the trained model are achieved; and the hidden layers, which are intermediate layers and can be zero, one or more. The connections between them are based on a weight update structure, i.e., values are altered through

model training until a stable framework with decision capacity is built (Martínez-Comesaña et al., 2021).

Having input dataset $X: X_1, \dots, X_{n0}$ and assuming that the neurons in each of the L layers are respectively $n_0, n_1, \dots, n_{L+1}$, the inputs of the first hidden layer have the shape presented by:

$$i_n = \sum_{j=1}^{n_0} w_{jn} x_j + b_n \tag{8}$$

where w_{jn} represents the weight value connecting the jth input neuron to the nth neuron in the first hidden layer and bn is the bias of the *n*th neuron in the first hidden layer. Feeding i_n into an activation function, the outputs of the first hidden layer follows:

$$o_n = f_1(i_n) \tag{9}$$

The same process is repeated for the other hidden layers, with $f_k(.)$ being the activation function of layer k, and at last, the inputs and outputs of the neurons in the output layer are obtained as exposed below for the *n*th neuron:

$$i_n = \sum_{j=1}^{n_0} w_{jn} o_j + b_n \tag{10}$$

$$o_n = i_n \tag{11}$$

The optimization of the weights is based on backpropagation training using a gradient descent algorithm. At each iteration, the Loss function is calculated with the mean square error. The workflow adopted as follows: (i) Analysis of different hyperparameter ranges and values on model accuracy; (ii) Establishment of the environment for Talos, that is, the specific ranges and values of each hyperparameter and structure of the network; (iii) Reasoning of state-of-the-art

results and exploration of a range of windows to filter the target variables, using the defined architectures; and (iv) Further adjustments to the network, for instance, a more detailed analysis of the number of neurons, culminating in optimal architectures.

Since NNs are prone to overfitting and problem-specific, it is necessary to have some idea of where the hyperparameters fall and how to structure the model to set adequate ranges to be tested. As the input data was transformed using PCA in buoy 41048, the scaling technique adopted was standardization, obligatorily. For the output, standardization and normalization transformations were tested, and the optimal strategy was found for the standardization of inputs and normalization of outputs.

After setting the hyperparameter ranges, which were rather meticulous, and including early stopping in the structure as the regularization method to avoid overfitting, the optimization process took about two days of computation. However, further testing showed that the model has the potential to improve results with a different regularization method. Initially, only early stopping was considered, which is the simplest form of regularization to avoid overfitting. It stops the training when the performance measure of choice stops improving, with the option to set a "patience" that delays the stopping by a specified number of epochs after no improvement. At first, strict patience of two epochs was set. It was concluded that such a method was too strict and other options would be beneficial. Thus, dropout was added to the model.

During training, dropout randomly ignores or "drop out" some number of layer outputs temporarily, making the layer look-like and be treated-like a layer with a different number of nodes and connectivity to the prior layer. In effect, each update to a layer during training is performed with a different "view" of the configured layer (Srivastava et al., 2014). A visual

example of the consequences of adding dropout to the network is presented in Figure 3. It has to be mentioned that without any regularization, overfitting would be guaranteed.

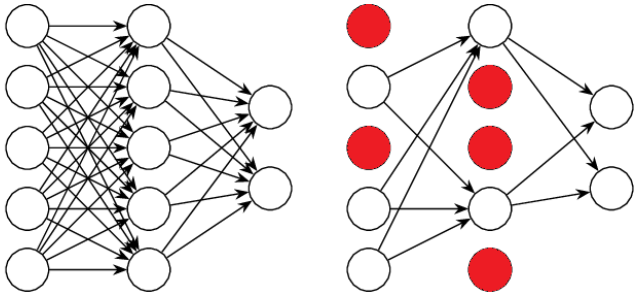

Figure 3 - Dropout effect. The left network is a standard, fully connected network. The right shows an example of the same network after applying dropout.

Finally, the structure and ranges of hyperparameters are defined: the activation function; the number of hidden layers and dropout regularization were fixed to single values, and the established ranges of the other hyperparameters are specified in Table 3. SGD is short for Stochastic Gradient Descend and the hyperparameters momentum and Nesterov are only fixed when combined with SGD. Kernel initializer defines the way to set the initial random weights of layers. The chosen activation is the popular ReLU, which is a common choice as it is both simple to implement and effective at overcoming the limitations of other previously popular activation functions, such as Sigmoid and Tanh. The dropout rate can be specified to each hidden layer as the probability of setting an input to the layer to zero, and it was verified that a value of 0.25|0.2 was sufficient. Early stopping was also applied but with a patience of 30|10 in buoy 41040|41048, preventing the model from training excessively as well as overfitting. This also allows the maximum number of epochs to be very large, without needing to tune. Less tolerance in the network for buoy 41048 derives from the increase in sensitivity. Achieving good enough results with the simplest model possible is a goal, so the models have only two hidden layers, and it is not only one because a larger network is indispensable as dropout will probabilistically

reduce the capacity of the network. Moreover, greater depth does improve generalization for the majority of tasks (Goodfellow et al., 2016). The shape of the NN was also defined upfront to be "brick", so both hidden layers have the same number of neurons.

The defined range of the number of neurons, in Table 3, is actually to obtain sub-optimal results, as this is a very important hyperparameter and setting an extensive range to run in Talos would exponentially increase complexity. The construction of the sub-optimal architectures was accomplished by interpreting the table of results, this is, loss and validation loss of each combination of parameters. Having such structures, the next step follows.

Table 3 - Hyperparameters, searched ranges, and resulting optimum values for MLP-NN architecture.

Hyperparameter	Range
Number of neurons	[5, 15, 50, 100, 200, 500]
Batch size	[1, 32, 64]
Optimizer	[Adam, SGD]
Kernel initializer	[Glorot uniform, Glorot normal, Orthogonal, He normal]
Learning rate	[10^{-3}, 10^{-2}, 10^{-1}]
Momentum	[0.99, 0.9, 0.5]
Nesterov	[True, False]

A problem initially found with the residues of swh and wspd is the excess of noise and outliers, which can expand the risk of overfitting (Krasnopolsky, 2014). This is a major difficulty for the optimization procedure used by the NN. Proper filtering of the time series can alleviate this problem. Figure 4 shows the residue of swh (e_swh, blue curve), where the level of noise and high-frequency fluctuations can be visualized, as well as an example of the filtered signal using a moving average of 24 hours (black curve).

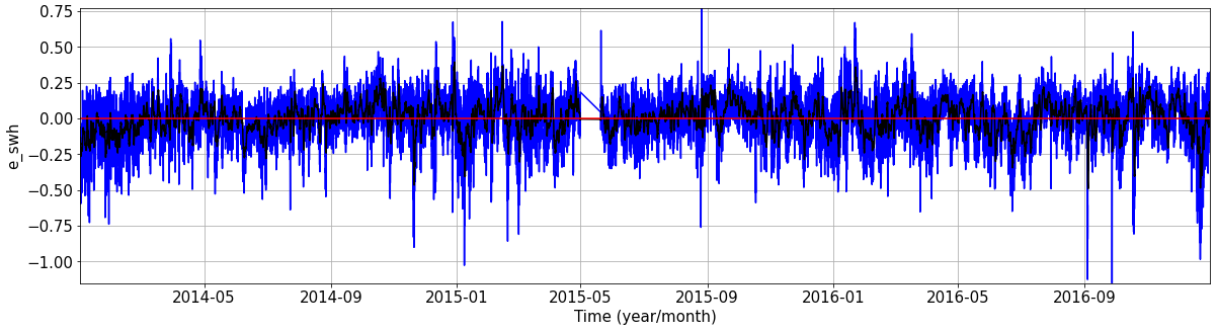

Figure 4 - Time series of the residue of swh (blue) and the filtered signal using a moving average of 24h (black).

As can be seen, a simple low-pass moving average filter can remove a great part of high-frequency oscillation that is mostly associated with random noise. However, it is not known a priori the size of the moving window that better optimizes the model. Therefore, NNs were built with the established sub-optimal architecture and trained on non-filtered - for reference - and filtered output variables with filtering windows of magnitudes 1, 3, 6, 12, 24 and 48h; each one trained five times to also evaluate the consistency of results. In total, 35 NNs were trained and analysed according to the evaluation metrics (Eqs. 4-7) on the adjusted signals. Such a strategy can be visualized in Figure 5 for buoy 41040, where the five results for each window are displayed as well as the mean value for each metric calculated. The optimal strategy for e_swh was found with a filtering window of 12h|1h in buoy 41040|41048, which becomes the new target of the NNs. Regarding e_wspd, in buoy 41040 no magnitude delivered better results than the original time series and, for buoy 41048, the best window size is 3h.

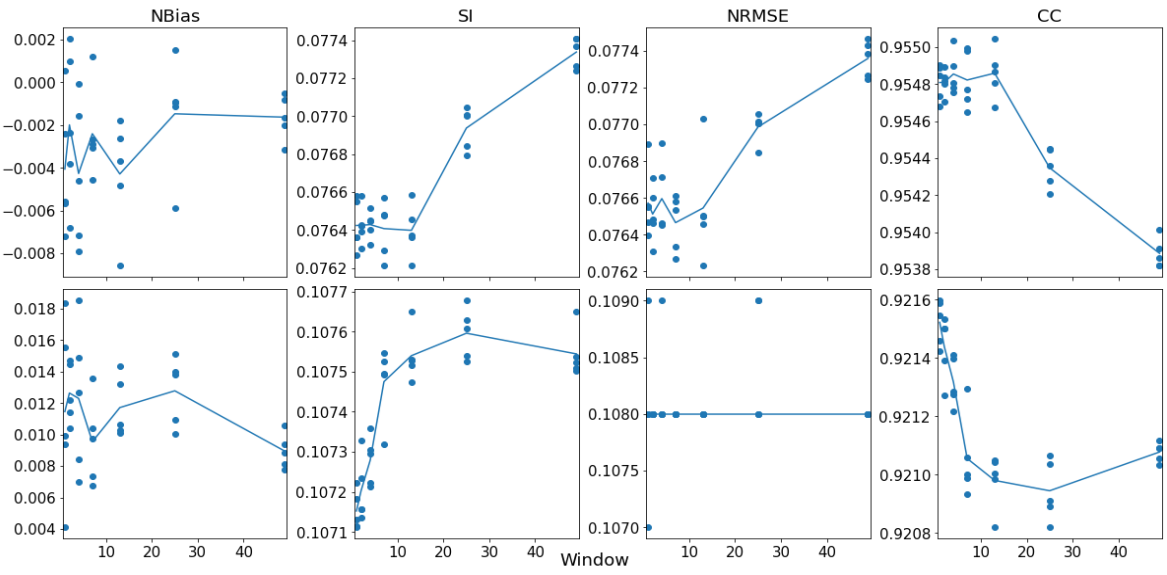

Figure 5 - Results of the Multi-output Neural Networks in buoy 41040 using the filtered signals as targets. The plots share the same x axis, which is the magnitude of the moving window. The magnitudes tested are 1, 3, 6, 12, 24 and 48h. Each horizontal set of plots exhibit the results of each variable, swh (top) and wspd (bottom) using the metrics NBias, SI, NRMSE and CC, from left to right, according to Eqs. 4-7.

As previously discussed, further adjustments to the architecture regarding the number of neurons were necessary. Having a multi-output model might mean that the general best approach is not the best for each target, and a trade-off is potentially beneficial. Therefore, analysing the prediction results individually is necessary. A total of 100 NNs were trained with the hyperparameters selected in the previous step and only varying the number of neurons: ranging from 5 to 305 with a step of 15, for 5 seeds to account for the randomness. Figure 6 shows the five results in buoy 41040 for each number of neuron as well as the mean value for each metric calculated. The optimal strategy for e_swh is different than for e_wspd, where for the first it corresponds to a value of 35 neurons per layer and for the second, 230. Considering that predicting e_wspd is more demanding, this trade-off was employed and thus 230 is the final optimal number of neurons per layer. Moreover, the predictions of e_swh do not decrease significantly when increasing the number of neurons. As for the results in buoy 41048, the optimal number of neurons for e_swh and e_wspd is 230 and 245, respectively, and the same

trade-off was employed although the disparity between the number of neurons is much less decisive. The optimal architectures are summarized in Table 4, where it can be seen that optimal networks are very similar for both buoys.

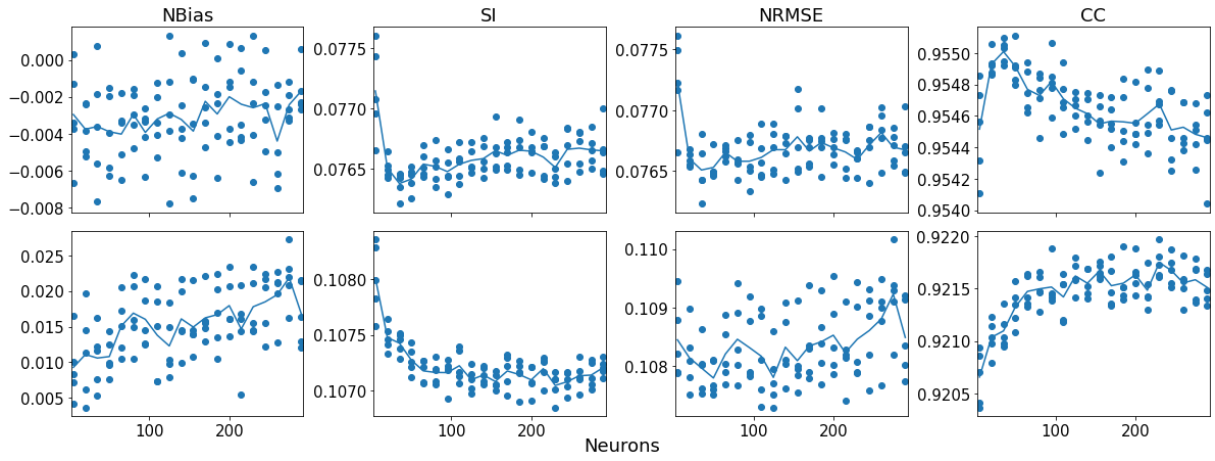

Figure 6 - Results of the Multi-output Neural Networks in buoy 41040 for different neurons. The plots share the same x axis, which is the number of neurons in a range from 5 to 305 with a step of 15. Each pair of vertical plots exhibit the results of each variable, swh (top) and wspd (bottom), using the metrics NBias, SI, NRMSE and CC, from left to right, according to Eqs. 4-7.

For the final result, the optimal model was run for 10 different seeds as means to account for randomness. After transforming the prediction results to the original scale and calculating the adjusted time series of the two variables of interest (difference between ERA5 estimates and predicted residues), the final averaged results are presented in Table 5, along with all other implementations and the initial reanalysis, as a means to examine the contrasts and compare them. Despite the efforts, the adjusted estimates with MLP presented a small improvement. Nonetheless, this is still a successful outcome since the reanalysis is already very accurate. Besides the bulk metrics, the next figures show additional benefits of using the MLP for post-processing.

To visually interpret the impact of adjustments, Figure 7 outlines part of 2016 curves of both targets in buoy 41040: measurement, era5 estimates, and adjusted estimates. The curves of swh

show evident improvements in most parts, whereas other parts are just very similar to the non-adjusted, while others, as expected, are slightly worse. One of the difficulties of reanalysis data is reproducing small-scale wave conditions. In this case, the MLP post-processing significantly improved the performance below 1.5 meters. As for the curves of wspd, the improvements are not so evident since the oscillations of the time series are more frequent and extreme. Still, NRMSE decreased from 0.1138 to 0.1084. The improvements in buoy 41048 were slightly greater.

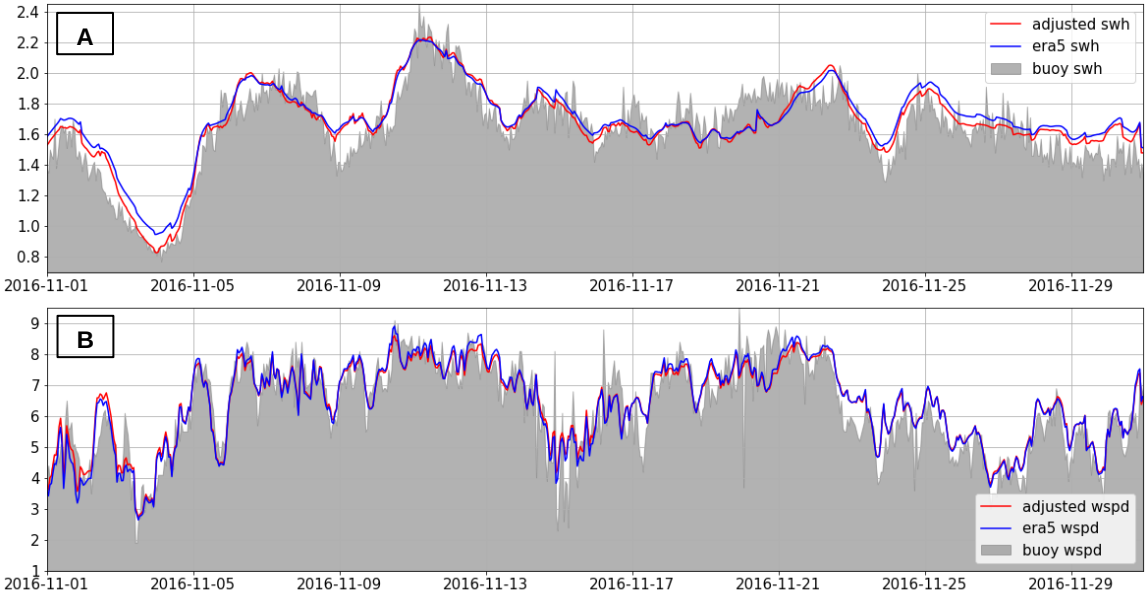

Figure 7 - Comparison between buoy measurements, ERA5 estimates, and adjusted estimates with the optimal MLP in buoy 41040. The two plots illustrate the time series of swh (a) and wpsd (b), zoomed to the test set's month that includes higher peaks.

3.2 Long Short-Term Memory (LSTM)

Long short-term memory (LSTM), proposed by Hochreiter and Schmidhuber (1997), is a type of recurrent neural network (RNN) used in the field of deep learning that allows the network to retain long-term dependencies at a given time from previous time steps. RNNs were designed specifically for the inclusion of feedback connections, unlike standard feedforward NNs.

However, long-term dependencies can make the network untrainable due to the vanishing gradient problem. LSTMs were developed precisely to solve that problem.

A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell. Figure 8 illustrates the diagram of a LSTM cell at the time step t. The description of the symbols and the network equations are:

- f - Forget Gate, responsible for removing information from the cell state.
- $\bar{C}$ - Candidate layer, contains all possible candidate values that could be added to the cell state.
- I - Input Gate, responsible for the addition of information to the cell state.
- O - Output Gate, decides what the next hidden state should be.
- H - Hidden state, contains information on immediately previous events.
- C - Cell state, long term memory of the model.
- W and U - Weight vectors of each gate.

Inputs to the LSTM cell at any step are X_t, H_{t-1} and C_{t-1}. Outputs of the LSTM cell are H_t and C_t.

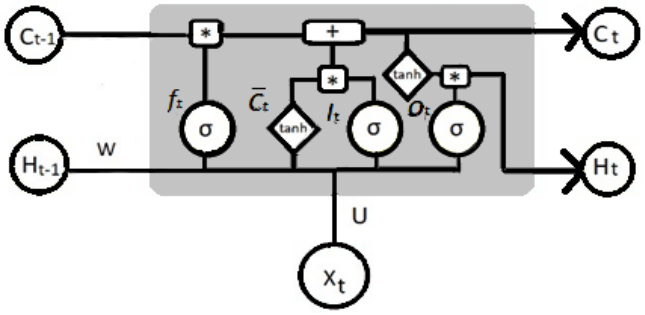

Figure 8 - Information flow in an LSTM block at time step t. The symbols * and + refer to element-wise multiplication and element-wise addition, respectively.

The three gates are the way to let information through. Each contains a sigmoid activation, which outputs numbers between 0 and 1. A result of 0 causes values to be "forgotten", while a result of 1 means that the values are "kept". This way the network learns which information is or isn't important. The default activation function for the candidate layer is tanh, which helps regulate the values flowing through the network, by squishing values to always be between -1 and 1. The formal network equations are given below.

$$f_t = \sigma\left(X_t * U_f + H_{t-1} * W_f + b_f\right) \tag{12}$$

$$\overline{C_t} = tanh(X_t * U_c + H_{t-1} * W_c + b_c) \tag{13}$$

$$I_t = \sigma(X_t * U_i + H_{t-1} * W_i + b_i) \tag{14}$$

$$O_t = \sigma(X_t * U_o + H_{t-1} * W_o + b_o) \tag{15}$$

$$C_t = f_t * C_{t-1} + I_t * \overline{C_t} \tag{16}$$

$$H_t = O_t * tanh(C_t) \tag{17}$$

These networks are well suited to make predictions based on time series data since there can be lags of unknown duration between important events. Maintaining temporality in the data is important for the network to learn patterns from the correct sequence of events. Therefore, the data should not be shuffled.

The input to every layer must be three-dimensional: number of samples, time steps, and features. Time steps are the chosen number of past events to learn from. Accordingly, the dataset must be prepared and transformed before feeding it to the NN. For further explanation, let's consider an input space X containing n features and an output space y with one variable to be predicted. The desired number of time steps is m. One sample of input will be $[X_1(t-1), \ldots, X_n(t-1), \ldots, X_1(t-m), \ldots, X_n(t-m)]$ and the target will be $y(t+p)$, p being the number of

time steps that represent a further out in the future prediction. Setting $p = 0$ means that the predictions are for the next step. The standard approach includes the target variable from previous time steps in the input space, for instance in univariate time series forecasting. A python function was built to address the reshaping of data where the number of previous and future time steps is chosen by the user.

Regarding the scaling of data, standardization was applied to both datasets as preliminary tests confirmed their suitability. As the output variables are included in the input space, these variables must be also standardized. Note that in MLP the output space was not standardized but normalized. Focusing on the same hyperparameters as for the MLP model, the sub-optimal choices are presented in Table 4 for both buoy locations. Because LSTMs are very complex and time-consuming Talos was not employed, which did not affect much as preliminary tests proved the efficiency of the models in sub-optimal environments. As the transformation of data is a necessary step before training the model, the arbitrary decision of selecting one previous time step served as core to the selection of hyperparameters, detailed below.

The search for a suitable filtering strategy was not as meticulous as with MLPs since, as previously mentioned, LSTMs are more complex and time-consuming. After experimentation, it was considered sufficient to apply a filter of magnitude 1h to the time series of e_swh and no filter to e_wspd for buoy 41040, and the exact opposite for the data from 41048.

Concerning the number of neurons, the approach employed for MLP algorithm was also employed for LSTM. Therefore, 100 LSTMs were trained with the hyperparameters chosen before, only varying the number of neurons, in a range from 5 to 305 and a step of 15, for 5 seeds to account for the randomness. Figure 9 shows the five results for each neuron as well as the average of each metric, for the data from buoy 41040. The opposite situation from before is

encountered as the optimal approach for e_swh contains much more neurons than for e_wspd, 200 and 80. Even though the prediction accuracy of wspd increases significantly with the present approach, it is still lower than of swh so the chosen number of neurons is 80 for the final architecture of LSTM. Besides, the curve of results in e_swh (Figure 9) does not suffer significant variations when increasing the number of neurons. In relation to buoy 41048, preliminary tests indicated the need for a larger range of neurons to test, so it was extended to 505. The optimal approach consists of 395 neurons and 65 for e_swh and e wspd, respectively. When expanding to additional time steps it was concluded that adding a fraction of the base number of neurons to the total number of neurons was beneficial in buoy 41040. Such an approach in buoy 41048 did not stimulate results so the number of neurons is constant for all datasets.

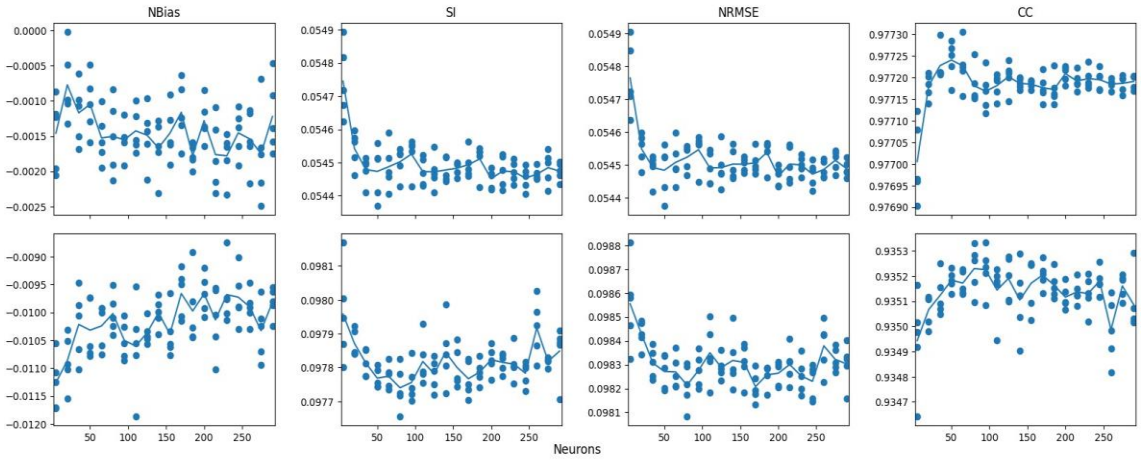

Figure 9 - Results of the Multi-output LSTM in buoy 41040 for different neurons. The plots share the same x axis, which is the number of neurons in a range from 5 to 305 with a step of 15. Each pair of vertical plots exhibit the results of each variable, swh (top) and wspd (bottom) using the metrics NBias, SI, NRMSE and CC, from left to right, according to Eqs. 1-4.

Regarding the regularization, it is not advisable to add dropouts to recurrent connections as LSTMs work well for long terms. However, an important feature about them is the lack of ability to "memorize" multiple characteristics simultaneously. The logic of dropout lies in adding noise

to the neurons in order not to be dependent on any specific neuron. But by adding dropout to recurrent connections, there is a chance of forgetting something that should not be excluded. In Zaremba et al. (2014) and Gal and Ghahramani (2016) it is exposed that applying dropout only to the nonrecurrent connections alone results in improved performance and successfully reduces overfitting. This was confirmed in the present study, and dropout was only optimized in the non-recurrent connections. The data from buoy 41040 is prone to overfitting in this environment, worsening as time steps are added to the input space. To adjust this situation, a base dropout rate of 0.2 was found to be suitable and a percentage corresponding to the number of time steps was added. For the same reason, patience was set to decrease as the time steps (feature space) increased, forcing the model to stop earlier. As for the data from buoy 41048, dropout was set to zero as the model generalizes well without overfitting. Patience was set to increase in accordance with the increase of dimension in feature space as it was verified that the model convergence requires more epochs. The weight initializers were set to the default values in Keras, Python.

Table 4 – Summary of the final architectures of both neural network models, MLP and LSTM, for both buoy IDs, 41040 and 41048. The structure of LSTM models is defined in the function of x, the number of previous time steps.

Model	**Neurons**		**HL**		**LR**		**Momentum**		**Patience**		**Dropout**		**WI**	
MLP	230	245	2	2	10^{-2}	10^{-2}	0.90	0.90	30	10	0.25	0	ort	ort
LSTM	[80(1+(x/10))]	65	1	1	10^{-2}	10^{-2}	0.95	0.90	30-x	20+x	0.2+x/100	0	def	def

After defining the general architectures, datasets including 1, 2, 3, 4, 5, 10 and 20 prior time steps were constructed. This is an interesting and important aspect to investigate. LSTMs were trained five times for each dataset, with different seeds, using the architectures summarized in Table 4. Results are displayed in Figure 10 for buoy 41040. It is evident that the efficiency of the model reaches a maximum between three- and five-time steps and from there it decreases,

denoting that there is no advantage in having a highly complex model. In fact, more than five days prior to the event negatively influences optimization, especially for e_wspd. A similar, yet inverted, situation is encountered for buoy 41048, as the excessive increase of time steps in input negatively affects more e_swh. Even though the two locations have distinct conditions, the efficiency of LSTM models is analogous in terms of the number of time steps. These results allow for the generalized conclusion that more than five days prior to an event should not be included as predictors.

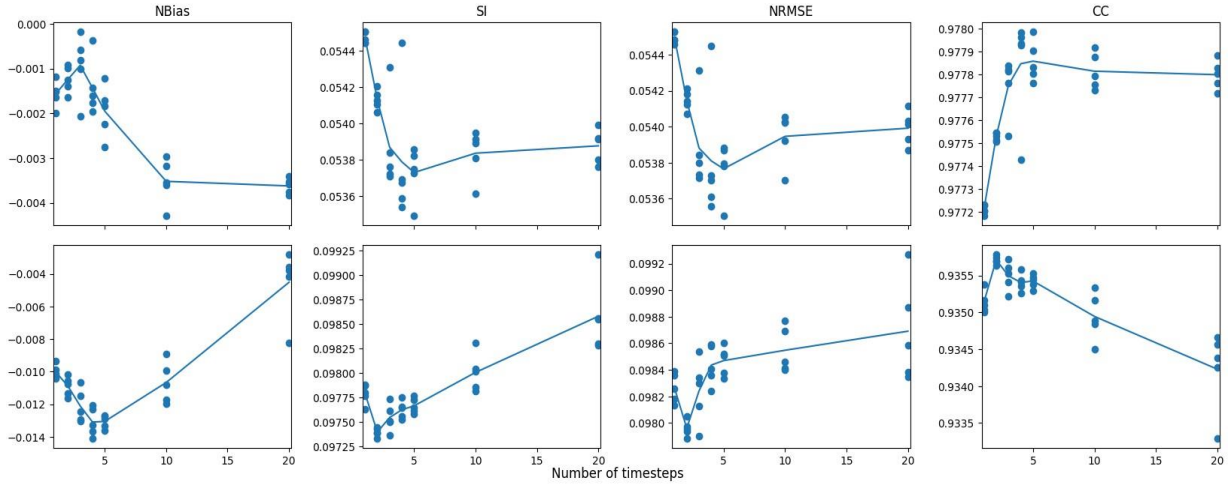

Figure 10 - Results of the Multi-output LSTM with different previous time steps for buoy 41040. The plots share the same x axis, which is the number of time steps. Each pair of vertical plots exhibit the results of each variable, swh (top) and wspd (bottom) using the metrics NBias, SI, NRMSE and CC, from left to right, according to Eqs. 4, 5, 6, 7.

Figure 11 depicts the best achievement, which is a LSTM with three prior time steps for buoy 41048. The month of February was selected for visualization as it is part of the test set with higher peaks. These curves, in the same manner as the curves in Figure 7 for MLP, overlap three time series: measurements, ERA5 estimates, and adjusted estimates. The model demonstrated the capability of predicting the residues very accurately to the point where the adjusted time series are very similar to the measurements, with minor differences. Thus, as the adjustments are

significantly improved, adopting such a strategy to the estimates could allow for similar error levels as observations – especially important when they are missing. As it can be seen in the boxplots of Figure 11, the residue data has many outliers, both the initial and the adjusted data. This indicates that extreme events with poor performance in the numerical models also show a worse performance with the NNs. One explanation for this deficiency in simulating large peaks (outliers) is that the infrequency of occurrence causes the training data for detecting outliers to be unavailable (Reunanen et al., 2020).

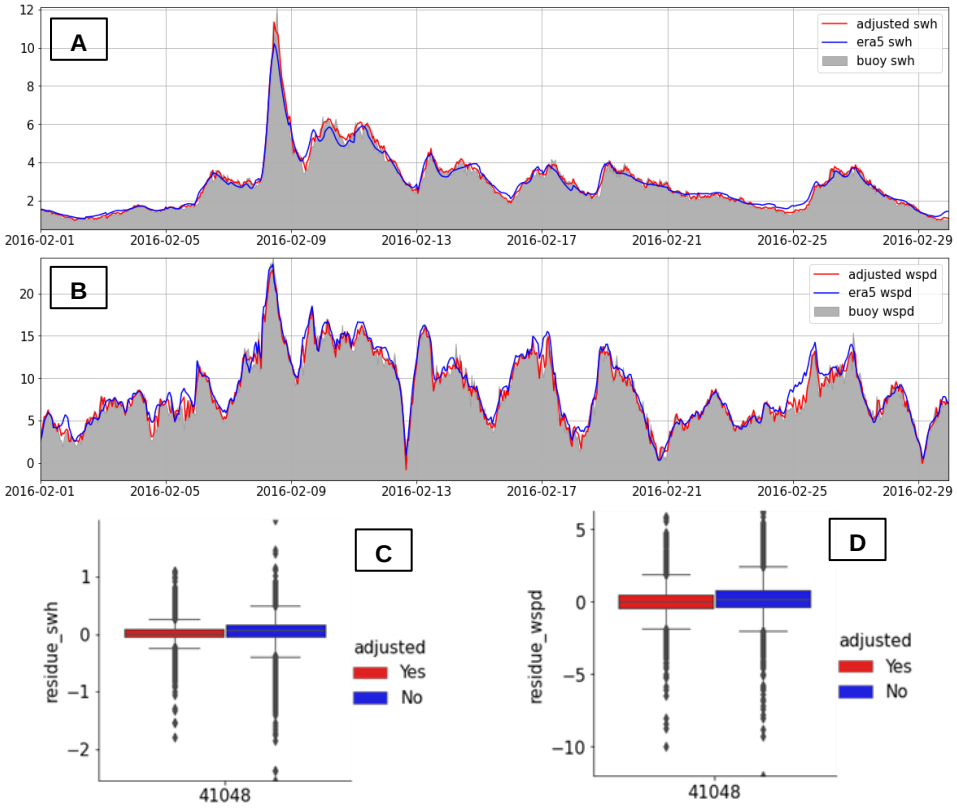

Figure 11 - Before and after results of the optimal LSTM for buoy 41048, having 3 time steps. The first two subfigures illustrate the curves of swh (a) and wpsd (b), zoomed to specific months of 2016 where peaks occurred. The two latter, (c) and (d), are the respective boxplots of the whole test set period, comparing the distribution of the residues before and after adjustments.

Table 5 - Error metrics of swh and wspd in 2016 for the reanalysis (ERA5) and the adjusted reanalysis using different NN architectures.

		41040 | 41048							
Model	Var.	NBias		SI		NRMSE		CC	
Reanalysis	*swh*	*0.001*	*0.019*	*0.081*	*0.117*	*0.081*	*0.118*	*0.953*	*0.974*
	wspd	*0.031*	*0.028*	*0.110*	*0.150*	*0.114*	*0.152*	*0.921*	*0.947*
MLP	swh	-0.002	0.023	0.077	0.111	0.077	0.113	0.955	0.974
	wspd	0.016	0.010	0.107	0.145	0.108	0.146	0.922	0.948
LSTM (1 step)	swh	-0.002	0.009	0.055	0.071	0.055	0.072	0.977	0.989
	wspd	-0.010	-0.001	0.098	0.129	0.098	0.129	0.935	0.959
LSTM (2 steps)	swh	-0.001	0.009	0.054	0.069	0.054	0.070	0.978	0.990
	wspd	-0.011	-0.001	0.097	0.128	0.098	0.128	0.936	0.960
LSTM (3 steps)	swh	-0.001	0.008	0.054	0.069	0.054	0.070	0.978	0.990
	wspd	-0.012	-0.002	0.098	0.126	0.098	0.126	0.936	0.961
LSTM (4 steps)	swh	-0.001	0.009	0.054	0.069	0.054	0.070	0.978	0.990
	wspd	-0.013	-0.001	0.098	0.126	0.098	0.126	0.935	0.961
LSTM (5 steps)	swh	-0.002	0.009	0.054	0.070	0.054	0.007	0.978	0.990
	wspd	-0.013	0.000	0.098	0.126	0.099	0.126	0.935	0.961
LSTM (10 steps)	swh	-0.004	0.008	0.054	0.071	0.054	0.071	0.978	0.990
	wspd	-0.011	0.003	0.098	0.126	0.099	0.126	0.935	0.961
LSTM (20 steps)	swh	-0.004	0.009	0.054	0.072	0.054	0.072	0.978	0.989
	wspd	-0.005	0.005	0.099	0.126	0.099	0.126	0.934	0.961

4. Conclusions

A large set of experiments is conducted to develop deep neural network-based models (DNN) to post-process and bias-correct ERA5 wind and wave data, using reliable buoy measurements in two locations of the North Atlantic Ocean presenting distinct metocean conditions. The main goal is to investigate two types of deep learning models able to emulate nonlinear data and effectively outperform the ERA5 reanalysis. Such types are MLP and LSTM models, where the output variables are the residues of significant wave height (swh) and wind speed (wspd), i.e., the difference between the reanalysis and the observations. A careful study

covering what environmental variables to include in the feature space is handled, as means to create datasets with a balance between efficiency and complexity and understand dependencies.

As expected, due to the memory feature of the nodes, LSTM models outperform MLP, even with suboptimal architectures. The best overall LSTM configuration is found to have five time steps and three time steps in input space, for buoys 41040 and 41048, respectively. It is shown that minimizing the residues of swh is more successful than of wspd, with the correlation coefficient between the adjusted values and the measurements of swh reaching 0.99, whereas the original ERA5 is 0.97. Additionally, the ERA5 RMSE of 8% and 12% for swh has been reduced to 5% and 7%, for buoys 41040 and 41048, respectively, with the LSTM post-processing model. The effects of such successful corrections are illustrated in Figure 11, which confirms the effectiveness of the presented models to reduce the systematic and scatter errors of the reanalysis.

Acknowledgments

This study has been performed under the project “WAVEFAI - Operational Wave Forecast using Artificial Intelligence”, which is funded by the Portuguese Foundation for Science and Technology (Fundação para a Ciência e a Tecnologia - FCT) under contract CIRCNA.OCT.0300.2019_1801P.01023. This work contributes to the Strategic Research Plan of the Center for Marine Technology and Ocean Engineering (CENTEC), which is financed by FCT under contract UIDB/UIDP/00134/2020. The second author was at CENTEC while this study was developed but is now funded by the Cooperative Institute for Marine and Atmospheric Studies (CIMAS), a Cooperative Institute of the University of Miami and the National Oceanic and Atmospheric Administration, cooperative agreement NA20OAR4320472. Acknowledgements are due to the European Centre for Medium-Range Weather Forecasts

(ECMWF) and the National Data Buoy Center (NDBC) for providing the data. The authors would like to thank Fabio Almeida for the great support with data management and computational issues.

References

Almeida, S.; Rusu, L., and Guedes Soares, C. 2016. Data assimilation with the ensemble Kalman filter in a high-resolution wave forecasting model for coastal areas. Journal of Operational Oceanography. 9(2)103-114.

Ahn, S., Tran, T. D., Kim, J., 2022. Systematization of short-term forecasts of regional wave heights using a machine learning technique and long-term wave hindcast, Ocean Engineering, 264, 112593.

Bai, G., Wang, Z., Zhu, X., Feng, Y. 2022. Development of a 2-D deep learning regional wave field forecast model based on convolutional neural network and the application in South China Sea, Applied Ocean Research, 118, 103012.

Blum, A. L., Langley, P. 1997. Selection of relevant features and examples in machine learning. Artificial Intelligence, 97(1-2), 245–271. https://doi.org/10.1016/S0004-3702(97)00063-5

Breivik, L.A. and Reistad, M. 1994. Assimilation of ERS-1 altimeter wave heights in an operational numerical wave model, Weather Forecasting, 9, 440-451.

Callens, A., Morichon, D., Abadie, S., Delpey, M., Liquet, B., 2020. Using Random forest and Gradient boosting trees to improve wave forecast at a specific location, Applied Ocean Research, 104, 102339.

Campos, R. M.; Costa, M. O.; Almeida, F., and Guedes Soares, C. 2021; Operational wave forecast selection in the Atlantic Ocean using Random Forests. Journal of Marine Science and Engineering . 9(3):298.

Campos, R. M., Guedes Soares, C. 2016. Comparison of HIPOCAS and ERA wind and wave reanalyses in the north Atlantic Ocean. Ocean Engineering, 112, 320–334. https://doi.org/10.1016/j.oceaneng.2015.12.028

Campos, R. M., Krasnopolsky, V., Alves, J.H.G.M, Penny, S.G., 2019. Nonlinear wave ensemble averaging in the Gulf of Mexico using neural networks. J Atmos Ocean Technol, 36, 113–127. https://doi.org/10.1175/JTECH-D-18-0099.1

Campos, R. M., Krasnopolsky, V., Alves, J.H.G.M, Penny, S.G, 2020. Improving NCEP's global-scale wave ensemble averages using neural networks. Ocean Modelling, 149:101617, https://doi.org/10.1016/j.ocemod.2020.101617

Dixit, P., Londhe, S. N., Dandawate, Y. H, 2014. Wave forecasting using neuro wavelet technique. The International Journal of Ocean and Climate Systems, 5(4):237–247, ttps://doi.org/10.1260/1759-3131.5.4.237

Domingos, P., 2012. A few useful things to know about machine learning. Communications of the ACM, 55(10):78–87.

Ebrahimi-Khusfi, Z., Nafarzadegan, A. R., Dargahian, F., 2021. Predicting the number of dusty days around the desert wetlands in southeastern iran using feature selection and machine learning techniques. Ecological Indicators, 125:10749. https://doi.org/10.1016/j.ecolind.2021.107499

Elbisy, M. S. and Elbisy A.M.S., 2021. Prediction of significant wave height by artificial neural networks and multiple additive regression trees Ocean Engineering, 230 109077.

Evensen, G.., 2003. The Ensemble Kalman Filter: theoretical formulation and practical implementation. Ocean Dynamics, 53: 343–367.

Fan, S., Xiao, N., Dong, S., 2020. A novel model to predict significant wave height based on long short-term memory network. Ocean Engineering, 205: 107298, https://doi.org/10.1016/j.oceaneng.2020.107298

Feng, X., Ma, G., Su, S.F., Huang, C., Boswell, M.K., Xue, P., 2020. A multi-layer perceptron approach for accelerated wave forecasting in Lake Michigan. Ocean Engineering, 211, 107526.

Gal, Y., Ghahramani, Z., 2016. A theoretically grounded application of dropout in recurrent neural networks. Advances in neural information processing systems, 29:1019–1027. https://doi.org/10.48550/arXiv.1512.05287

Galanis, G.., Emmanouil, G.., Chu, P., Kallos, G.., 2009. A new methodology for the extension of the impact of data assimilation on ocean wave prediction. Ocean Dynamics, 59:523-535.

Gao, Z., Liu, X., Yv, F., Wang, J., Xing, C., 2023. Learning wave fields evolution in North West Pacific with deep neural networks, Applied Ocean Research, 130, 103393.

Gemmrich, J., Thomas, B., Bouchard, R., 2011. Observational changes and trends in northeast pacific wave records. Geophysical Research Letters, 38(22), https://doi.org/10.1029/2011GL049518

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press, http://www.deeplearningbook.org.

Gurney, K., 2018. An introduction to neural networks. CRC press. ISBN 0-203-45151-1

Guyon, I., Weston, J., Barnhill, S., Vapnik, V., 2002. Gene selection for cancer classification using support vector machines. Machine Learning, 46(1):389–422, https://doi.org/10.1023/A:1012487302797

Han, Y., Zhang, Q., Li, V. O., Lam, J. C., 2021. Deep-air: A hybrid CNN-LSTM framework for air quality modeling in metropolitan cities. arXiv preprint arXiv:2103.14587, https://doi.org/10.48550/arXiv.2103.14587

Hersbach, H., Bell, B., Berrisford, P., Hirahara, S., Horányi, A., Muñoz-Sabater, J., Nicolas, J., Peubey, C., Radu, R., Schepers, D., Simmons, A., Soci, C., Abdalla, S., Abellan, X., Balsamo, G., Bechtold, P., Biavati, G., Bidlot, J., Bonavita, M., De Chiara, G., Dahlgren, P., Dee, D., Diamantakis, M., Dragani, R., Flemming, J., Forbes, R., Fuentes, M., Geer, A., Haimberger, L., Healy, S., Hogan, R. J., Hólm, E., Janisková, M., Keeley, S., Laloyaux, P.,

Lopez, P., Lupu, C., Radnoti, G., de Rosnay, P., Rozum, I., Vamborg, F., Villaume, S., and Thépaut, J. N., 2020. The ERA5 Global Reanalysis. Q. J. R. Meteorol. Soc., 146(730), 1999–2049, https://doi.org/10.1002/qj.3803

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural computation, 9(8):1735–1780, https://doi.org/10.1162/neco.1997.9.8.1735

Holland, S. M., 2008. Principal components analysis (PCA). Department of Geology, University of Georgia, Athens, GA, 30602–2501.

Hornik, K., Stinchcombe, M., White, H., 1989. Multilayer feedforward networks are universal approximators. Neural Networks, 2(5):359–366, https://doi.org/10.1016/0893-6080(89)90020-8

Huang, L., Jing, Y., Chen, H., Zhang, L., Liu, Y., 2022. A regional wind wave prediction surrogate model based on CNN deep learning network, Applied Ocean Research,126, 103287.

James, S.C., Zhang, Y., O'Donncha, F., 2018. A machine learning framework to forecast wave conditions. Coastal Eng. 137, 1–10.

Jing, Y., Zhang, L., Hao, W., Huang, L., 2022.Numerical study of a CNN-based model for regional wave prediction, Ocean Engineering, 255, 111400.

Jörges, C., Berkenbrink, C., Stumpe, B., 2021. Prediction and reconstruction of ocean wave heights based on bathymetric data using LSTM neural networks. Ocean Eng. 232, 109046, https://doi.org/10.1016/j.oceaneng.2021.109046

Kagemoto, H., 2020, Forecasting a water-surface wave train with artificial intelligence- A case study, Ocean Engineering, 207, 107380.

Katrutsa, A., Strijov, V., 2017. Comprehensive study of feature selection methods to solve multicollinearity problem according to evaluation criteria. Expert Systems with Applications, 76:1–11, https://doi.org/10.1016/j.eswa.2017.01.048

Krasnopolsky, V., 2014. NCEP neural network training and validation system: Brief description of NN background and training software. Environment Modeling Center, NCEP/NWS, NOAA, https://www.lib.ncep.noaa.gov/ncepofficenotes/files/on478.pdf

Krasnopolsky, V., Schiller, H., 2003. Some neural network applications in environmental sciences Part I: forward and inverse problems in geophysical remote measurements. Neural Netw. 16, 321–334.

Langley, P, 1994. Selection of relevant features in machine learning. In Proceedings of the AAAI Fall symposium on relevance, volume 184, pages 245–271, https://www.aaai.org/Papers/Symposia/Fall/1994/FS-94-02/FS94-02-034.pdf

Li, Y., Li, T., Liu, H., 2017. Recent advances in feature selection and its applications. Knowledge and Information Systems, 53(3):551–577. https://doi.org/10.1007/s10115-017-1059-8

Lionello P., H. Günter, and P. A. E. M. Janssen, 1992: Assimilation of altimeter data in a global third-generation wave model. J Geophys Res, 97(C9): 14, 463–474.

Londhe, S.N., Shah, S., Dixit, P.R., Nair, T.M.B., Sirisha, P., Jain, R., 2016, A Coupled Numerical and Artificial Neural Network Model for Improving Location Specific Wave Forecast, Applied Ocean Research, 59, 483-491.

Ma, X., Duan, W., Huang, L., Qin, Y., Yin, H., 2022. Phase-resolved wave prediction for short crest wave fields using deep learning, Ocean Engineering, 262, 112170.

Martínez-Comesaña, M., Ogando-Martínez, A., Troncoso-Pastoriza, F., López-Gómez, J., Febrero Garrido, L., Granada-Alvarez, E., 2021. Use of optimised mlp neural networks for spatiotemporal´ estimation of indoor environmental conditions of existing buildings. Building and Environment, 205:108243. https://doi.org/10.1016/j.buildenv.2021.108243

Menéndez, M., Méndez, F. J., Losada, I. J., Graham, N. E., 2008. Variability of extreme wave heights in the northeast pacific ocean based on buoy measurements. Geophysical Research Letters, 35 (22), https://doi.org/10.1029/2008GL035394

Mentaschi, L, Besio, G, Cassola, F, Mazzino, A, 2013. Problems in RMSE-based wave model validations. Ocean Modelling 72:53–58. https://doi.org/10.1016/j.ocemod.2013.08.003

Ni, C., Ma, X., 2020. An integrated long-short term memory algorithm for predicting polar westerlies wave height, Ocean Engineering, 215, 107715.

Oo, Y. H., Zhang, H., 2022. Spatial wave assimilation by integration of artificial neural network and numerical wave model, Ocean Engineering, 247, 110752.

Paul, R. K., 2006. Multicollinearity: Causes, effects and remedies. IASRI, New Delhi, 1(1):58–65.
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.498.1478&rep=rep1&type=pdf

Pirhooshyaran, M., Snyder, L. V., 2020. Forecasting, hindcasting and feature selection of ocean waves via recurrent and sequence-to-sequence networks, Ocean Engineering, 207, 107424.

Qiao, B., Wu, Z., Tang, Z., Wu, G., 2021. Sea surface temperature prediction approach based on 3d cnn and lstm with attention mechanism. In 2021 23rd International Conference on Advanced Communication Technology (ICACT), pages 342–347. IEEE.

Rao, S., Mandal, S., 2005. Hindcasting of storm waves using neural networks. Ocean Engineering, 32 (5-6):667–684. https://doi.org/10.1016/j.oceaneng.2004.09.003

Reunanen, N., Raty, T., Jokinen, J. J., Hoyt, T., Culler, D., 2020. Unsupervised online detection and prediction of outliers in streams of sensor data. International Journal of Data Science and Analytics, 9(3):285–314. https://doi.org/10.1007/s41060-019-00191-3

Rumelhart, D. E., Hinton, G. E., Williams, R. J., 1985. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science.
https://web.stanford.edu/class/psych209a/ReadingsByDate/02_06/PDPVolIChapter8.pdf

Rusu, L., Guedes Soares, C., 2014. Local data assimilation scheme for wave predictions close to the Portuguese ports. J. Operational Oceanogr.7(2), pp. 45–57.

Rusu, L. and Guedes Soares, C. 2015. Impact of assimilating altimeter data on wave predictions in the western Iberian coast. Ocean Modelling. 96:126-135

Segreto, T., Simeone, A., Teti, R., 2014. Principal component analysis for feature extraction and NN pattern recognition in sensor monitoring of chip form during turning. CIRP Journal of Manufacturing Science and Technology, 7(3):202–209. https://doi.org/10.1016/j.cirpj.2014.04.005

Skittides, C., Fruh, W.-G., 2014. Wind forecasting using principal component analysis. Renewable Energy, 69:365–374. https://doi.org/10.1016/j.renene.2014.03.068

Sophian, A., Tian, G. Y., Taylor, D., Rudlin, J., 2003. A feature extraction technique based on principal component analysis for pulsed eddy current ndt. NDT & e International, 36(1):37–41. https://doi.org/10.1016/S0963-8695(02)00069-5

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., 2014. Dropout: a simple way to prevent neural networks from overfitting. The journal of Machine Learning Research, 15 (1):1929–1958. https://jmlr.org/papers/v15/srivastava14a.html

Stopa, J. E., Cheung, K. F., 2014. Intercomparison of wind and wave data from the ecmwf reanalysis interim and the ncep climate forecast system reanalysis. Ocean Modelling, 75:65–83. https://doi.org/10.1016/j.ocemod.2013.12.006

Voorrips, A.C., Heemink, A.W., Komen, G.J., 1999. Wave data assimilation with the Kalman filter. Journal of Marine Systems. 19, 267-291

Wahle, K., Staneva, J., Guenther, H., 2015. Data assimilation of ocean wind waves using Neural Networks. A case study for the German Bight, Ocean Modelling, 96, 117–125

Wei, Z., 2021. Forecasting wind waves in the US Atlantic Coast using an artificial neural network model: Towards an AI-based storm forecast system, Ocean Engineering, 237, 109646.

Zamani, A., Azimian, A., Heemink, A., Solomatine, D., 2010. Non-linear wave data simulation with an ANN-type wind-wave model and Ensemble Kalman filter (ENKF), Appl. Math. Modell. 34 (8), 1984–1999.

Zaremba, W., Sutskever, I., Vinyals, O., 2014. Recurrent neural network regularization. arXiv preprint arXiv:1409.2329. https://doi.org/10.48550/arXiv.1409.2329

Zhao, Z., Morstatter, F., Sharma, S., Alelyani, S., Anand, A., Liu, H., 2010. Advancing feature selection research. ASU feature selection repository, 1–28. https://www.public.asu.edu/~huanliu/papers/tr-10-007.pdf

Zheng, Z., Ma, X., Huang, X., Ma, Y., Dong, G., 2022. Wave forecasting within a port using WAVEWATCH III and artificial neural networks, Ocean Engineering, 255, 111475.

Zhou, S., Xie, W., Lu, Y., Wang, Y., Zhou, Y., Hui, N., Dong, C., 2021. ConvLSTM-based Wave Forecasts in the South and East China Seas. Front Mar Sci 8, 740.