MASTER

Investigation into deep learning frameworks to forecast the significant wave height

van den Eijnde, F.

Award date:
2022

TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY
Department of Mathematics and Computer Science
Uncertainty in Artificial Intelligence

FUGRO
Innovation Department
Metocean Group

Investigation into deep learning frameworks to forecast the significant wave height

Master Thesis Report

Fenno van den Eijnde

Graduation committee:	dr. ir. E. Quaeghebeur
	N. Sepasian, Phd.
	prof. dr. ir. E. van den Heuvel
TU/e supervisor:	dr. ir. E. Quaeghebeur
Fugro Ssupervisors:	N. Sepasian, Phd.
	J. Liria Fernandez, MSc.

September 7, 2022

Abstract

Accurate sea state predictions are essential for offshore operations' safety and indicating vessels' operability. A crucial parameter to indicate the sea state is the significant wave height (SWH). In recent years, deep learning models have shown an advantage over numerical models in forecasting such weather conditions. These deep models need less computational power and no specific domain knowledge. However, these models require sufficient training data, which is not always available. Previous work has failed to address the difficulty of forecasting the SWH while limited data is available. Furthermore, they failed to address the usability of obtained accuracy. In this work, we comprehensively analysed the use of recurrent deep models by experimenting with different architectures and parameter settings. In our experiments, no models were found that could produce an usable SWH forecast. To enhance the predictive capabilities of these models, we explored transfer learning. This was not feasible with the current state of deep models for SWH forecasting. Finally, we proposed two frameworks which use empirical mode decomposition (EMD). This decreases the complexity of the data and therefore increases the interpretability for the models. However, both methods were unable to increase the models' performance. This thesis provides future studies a starting point and research directions for forecasting the SWH using deep learning models.

Contents

List of Figures

List of Tables

Acronyms

ACF autocorrelation function. 4, 5, 31, 33–35, 68

ADF augmented Dickey-Fuller. 31

APD average wave period. 18, 33, 61, 62, 64

AR autoregressive. 6, 25, 28, 34, 36, 37

CEEMD complete ensemble empirical mode decomposition. 20, 59

CNN convolutional neural networks. 18

DBN dynamic Bayesian network. 4, 17, 18

EEMD ensemble empirical mode decomposition. 20, 59

ELM extreme learning machine. 17

EMD empirical mode decomposition. 1, 5, 6, 11–13, 20, 21, 49–59, 75, 76

FPSO floating production, storage and offloading. 10

GRU gated recurrent unit. 4, 6, 13, 17, 27, 28, 34, 37, 38, 40, 41, 58

GST gust speed. 18, 33, 61, 64

IMF intrinsic mode function. 6, 12, 20, 21, 24, 49–51, 53, 54, 57, 59, 75

LSTM long short term memory. 4, 6, 7, 13, 17, 19, 26–28, 34, 37, 38, 40, 41, 43–47, 53, 57–59, 70

MLP multilayer perceptron. 4, 16, 17

MSE mean squared error. 35

NBDC National Buoy Data Center. 23

NLP Natural language processing. 11, 18, 35

NN neural network. 11, 15, 17–19, 25, 26, 28, 31, 53

PACF partial autocorrelation function. 4, 31, 33–35

PCC Pearson correlation. 5, 6, 13, 29, 31, 36–41, 43, 46–48, 55, 58, 69–73

ResNet residual neural network. 17

RF random forest. 17, 18

RMSE root mean squared error. 5, 13, 29, 36–41, 46–48, 55, 58, 69–73

RNN recurrent neural network. 4, 6, 13, 17, 25, 26, 28, 29, 34, 37, 38, 40, 41, 58

sMAPE symmetric mean absolute percentage error. 5, 13, 19, 29, 36–41, 46–48, 55, 58, 69–73

SVM support vector machine. 17, 18

SWH significant wave height. 1, 4–7, 10–15, 17, 18, 20, 22–25, 29–33, 35, 36, 45, 46, 48, 49, 51–53, 55, 57–59, 61, 62, 64, 67, 68, 75

WSPD wind speed. 18, 33, 61

Chapter 1

Introduction

In this thesis, we studied deep learning frameworks to forecast the significant wave height (SWH). This chapter introduces the research field of forecasting the SWH, outlines the research performed and summarises the findings. First, the context around forecasting the SWH is discussed in section 1.1. The literature in this field is briefly discussed in section 1.2. In section 1.3 the specific goals in this thesis are addressed, and in section 1.4 the research approach is summarised. Finally, the results and findings are summarised in section 1.5.

1.1 Context and topic

For offshore operations, efficient and good planning are essential to ensure personal, environmental and equipment safety. The ability to plan marine operations is decisive for the economic viability of any project. Therefore, strategies and methodologies that improve these operations' planning are of interest to increase safety and financial feasibility. Offshore operations frequently encounter hazardous and dangerous conditions caused by heavy equipment, dangerous materials, and treacherous weather [2, 3]. Weather conditions have a significant impact since these cannot be controlled and influence the safety of offshore operations. Therefore weather conditions are a crucial aspect of whether and when specific operations can be performed. A 5 to 7-day forecast window is usually necessary for safe and efficient planning of operations.

An example of an offshore operation would be to operate a floating production, storage and offloading (FPSO), a vessel used by the offshore oil and gas industry. This vessel cannot be operated under certain weather conditions and needs to evacuate to the closest harbour. Disconnecting and evacuating this platform takes approximately two days, meaning a weather forecast is necessary. Evacuating early causes unnecessary operational and downtime costs, while evacuating late could be disastrous for personal and equipment safety. For small construction and maintenance operations, e.g. wind farms, more short-term forecasting horizons are desired. These operations are executed by smaller vessels, which could abort and start operations on short notice.

Coastal areas influenced by coastal waves could also benefit from these weather forecasts [4]. Many coastlines worldwide are protected by coastal defence systems such as dikes. These systems attenuate waves and reduce the likelihood and volume of waves passing [5]. Waves passing these systems could lead to dangerous situations, such as sweeping people into the oceans or damaging coastal constructions. Accurate forecasts could give early warnings and prevent accidents from happening.

To identify hazardous periods, multiple factors of the weather conditions must be considered, e.g. the wind, the waves and the temperatures. With some criteria for these factors, a decision can be made whether an operation is feasible in a specified time slot. One crucial factor to evaluate is the prognosis of wave heights, or more specifically, the SWH. Since waves constantly vary in length, height and period, the SWH is used. The SWH at a location is a parameter approximately equal to the highest one-third of all waves passing. The SWH has proved to be an important indicator for safe and successful offshore operations.

To produce a forecast of the SWH, nowadays numerical models are used (e.g. WAM [6] and SWAN) [7]). These models use a grid of local measurements to simulate the underlying physics. However, these physical processes, especially wave generation by wind, are uncertain, complex, non-linear and non-stationary, increasing the difficulty of forecasting with numerical models [8]. In summary, these models are complex, time-consuming, and not always reliable, and there is a necessity for domain knowledge [9].

In recent years, machine learning and deep learning techniques have increasingly been used to forecast numerous time series, including SWH predictions, e.g. [10, 11, 12]. These models generally require fewer input variables, do not need domain-specific knowledge, require less computational effort and are still able to find complex patterns. Therefore, these models can be efficiently used for different geographical locations.

A disadvantage of these deep models is the requirement for a considerable amount of historical data for training. However, this is not always available for specific locations where operations are planned. This lack of data would limit the use of deep models by decreased accuracy. However, some methods are known to assist a deep model once limited data is available. In this thesis, we are interested in forecasting the SWH for locations with such limited data. A visual representation of an example of this problem is given in the fig. 1.1. Four buoys can be seen in the figure, where we would be interested in producing a forecast for the buoy at location B-3.

1.2 State of the art for forecasting the significant wave height

This section summarises the current state of the art for SWH forecasting, details about literature can be found in section 2.2.

In recent years, there have been a rising number of studies in forecasting the SWH using deep models. Especially many papers are published using neural networks (NNs) [8, 14, 15, 9, 16, 17]. However, NNs assume independence of input variables, which is not valid for time series. Sequential deep models are able to process time series and seem to outperform the NNs [18, 19, 20]. In the literature, predominantly univariate models are used for SWH forecasting. However, there is a variety of geographical and meteorological data available. In Li and Liu [21], they leverage this data by causal modelling and obtain similar results as the deep sequential models.

A well-known technique in computer vision and Natural language processing (NLP) to overcome limited data is transfer learning, where data is leveraged from a source dataset. This particular technique could be interesting, since data is available at nearby buoys. However, this technique applied to time series is a niche in the literature and treacherous since accuracies could drop [22, 23, 23].

In the literature various papers apply empirical mode decomposition (EMD) to in-

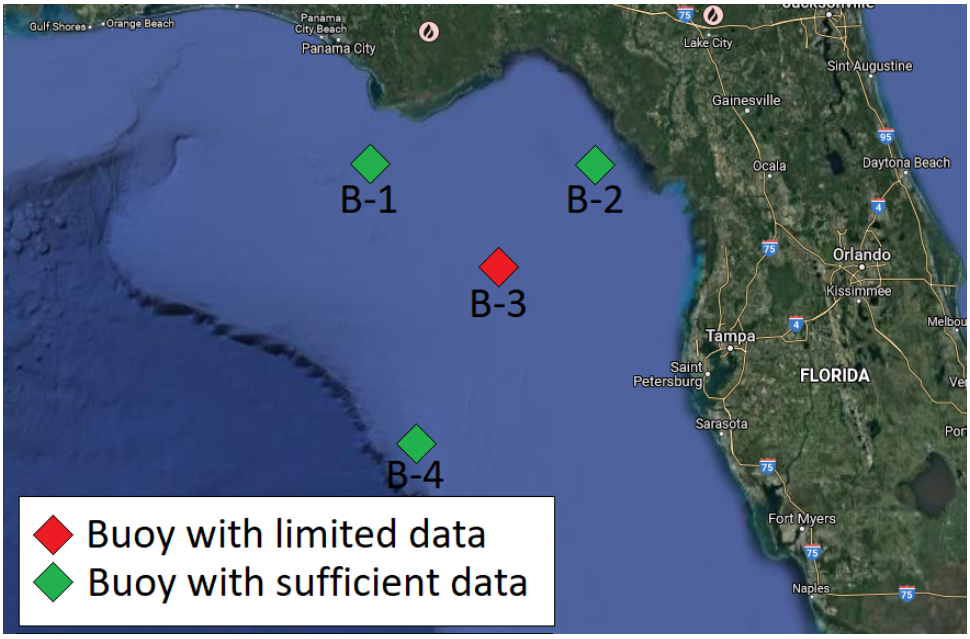


Figure 1.1: An example of four buoys in the Gulf of Mexico, only a limited dataset is available at B-3. Where we would be interested in creating a model to forecast the SWH at B-3. The figure is adapted from [13]

crease the forecast accuracy of the deep recurrent models [24, 25, 26, 27, 28, 29, 30]. EMD is a method that decomposes the signal in multiple intrinsic mode functions (IMFs), these IMFs are then individually forecasted and summed for an SWH forecast. However, we must note that these papers use decomposition before splitting the dataset into training and test sets. Therefore information is leaked from the test set to the training set.

1.3 Problem definition and research questions

In the current literature, numerous papers have been published discussing the use of deep sequential models to forecast the SWH. These papers show and discuss the results these models could achieve. However, all these papers assume sufficient data to train the models. There is a gap in the literature concerning the forecasting of the SWH while using limited datasets. In this thesis, we investigate using deep models while limited data is available. A formal description of this problem can be found in section 3.3. We investigate two methods to use deep recurrent networks at locations where only a limited amount of historical SWH measurements are available. The problem definition leads to the following research questions with sub-questions:

- Can transfer learning improve forecasts for the significant wave height at a location where limited data is available?
    - What geographical, meteorological features indicate which source buoys are suitable to transfer information for forecasting the significant wave height?
- Can empirical mode decomposition be used in a machine learning framework to enhance the forecasting of the significant wave height?

1.4 Method and approach

Here we briefly discuss the methods and approaches used in this research. Details about the methods and approaches can be found in the relevant chapters.

To investigate any enhancement techniques for deep learning frameworks, we first set a baseline for the performance of deep models in chapter 4. Two buoys, 414046 and 414047, located east of the Bahamas, are used for all experiments. These datasets are preprocessed and exploratorily analysed in section 4.2, which enables us to determine (hyper-) parameters for the architectures and training methods.

In section 6.2, three different deep recurrent models are proposed to forecast the SWH: a recurrent neural network (RNN), a gated recurrent unit (GRU) and a long short term memory (LSTM) network. Experiments are run using various model builds and hyperparameters to determine the effectiveness of different models and their parameters. In total 720 experiments are run for each buoy, which are evaluated compared to a baseline using the root mean squared error (RMSE), the symmetric mean absolute percentage error (sMAPE) and the Pearson correlation (PCC).

Once the baseline of the deep models is known, we consider transfer learning in chapter 5. This method is applied to overcome a lack of training data. In section 5.1, we discuss three different methods to transfer weights between the models. The models are trained on a source buoy, where we freeze and fine-tune various weights to forecast the target buoy.

In chapter 6, we investigate the possibilities of using EMD to enhance model performance. And we propose two frameworks which use EMD in a deep learning framework in section 6.2.

1.5 Findings

The results for the established baseline for the deep recurrent models are shown and interpreted in sections 4.4 and 4.5. The deep models could produce a slight improvement over the baseline's performance. However, these improvements were negligible, and the models learn behaviour similar to this baseline. The models could not produce a feasible forecast for the SWH.

The results for transfer learning are shown and interpreted in sections 5.2 and 5.3. No improvements were obtained by applying transfer learning, and there was no change in the forecasting behaviour. The models could learn the same behaviour with only two months of data, implying that the amount of data is not a problem to forecast the SWH.

The results for the deep EMD frameworks are shown and interpreted in sections 6.3 and 6.4. Both frameworks decreased the accuracy and could not produce a valid forecast for the SWH. The set baseline outperformed both frameworks, rejecting the usability of the proposed frameworks.

Chapter 2

Background

This chapter contains the preliminary theory necessary to understand the context of this work. First, in section 2.1 basic concepts are explained. And in section 2.2 the related work, including the state-of-the-art, is discussed.

2.1 Description of essential concepts

In this section, the significant wave height (SWH) is discussed in more depth to give insights into what we are trying to forecast. Furthermore, the concept of artificial neurons and the networks that can be constructed with them are explained.

2.1.1 Significant wave height

The SWH (H_s) characterises the sea state, including winds and swell. The sea state is the general condition of the surface over a large body of water at a certain time and location. This state is predominantly determined by wind waves and swells, a series of mechanical waves propagating over the surface under the influence of gravity. The SWH is a mathematical expression to estimate the average wave height for an observer at sea [31]. Originally the SWH was defined as the average of the highest one-third of all passing waves; in practice, this only differs a few percentages compared to the current definition. Which is defined as

$$H_s = 4 \cdot \sqrt{m_0}, \tag{2.1}$$

where m_0 is the wave displacement's variance, which is four times the surface elevation's standard deviation [32]. m_0 can be calculated by

$$m_0 = \sum_{f_l}^{f_u} S(f) \cdot d(f), \tag{2.2}$$

where the sum of spectral density, $S(f)$, is the over all observed frequency bands, from the lowest frequency f_l to the highest frequency, f_u, of the non-directional wave spectrum. And $d(f)$ is the bandwidth of each observed band. Note that if a continuous signal is used an integral is necessary to calculate m_0.

The SWH approximately follows the Rayleigh distribution [33]. Figure 2.1 shows a Rayleigh distribution of waves and the derivation of SWH from this distribution. Note that the SWH is significantly higher than the mean of all incoming waves.

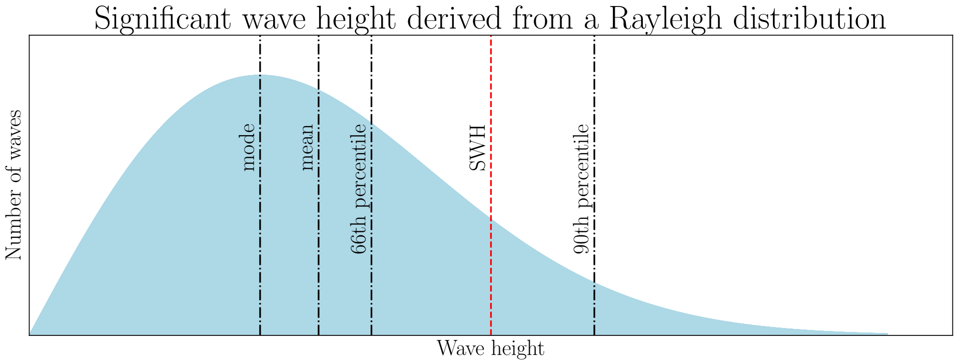


Figure 2.1: The statistical distribution of ocean wave heights, approximated by the Rayleigh distribution. The dashed red line indicates the SWH, which is the fourth line from the right.

2.1.2 Neural network

A neural network (NN) is a machine learning framework inspired by the functioning and structure of the human brain. This network consists of multiple artificial neurons, which transform (multiple) inputs into an output. The inputs are multiplied by weights $\mathbf{w}$ to create a calculated sum of the inputs, which is then offset by a bias b term. Each neuron has its own weights and biases, altogether denoted by θ.

Over this weighted sum the neuron applies an activation function ϕ to produce an output $o_\theta(x)$. This ϕ can be any activation function but is usually non-linear (e.g. tanh, sigmoid or softmax). Every single neuron computes

$$o_\theta(x) = \phi\left(\sum_i w_i x_i + b\right) = \phi\left(\mathbf{w}^T\mathbf{x} + b\right). \tag{2.3}$$

Given a certain input, this computation is graphically represented in fig. 2.2.

The network must find the "best" values for the neuron parameters θ to produce good outcomes. To find these parameters, the model will adjust its parameters during training to minimise the error specified by a loss function. The loss function is a function over the actual and the predicted outcomes $L(y^i, \hat{y}^i)$. This will indicate how far the network's prediction is off. Therefore, a suitable loss function should be chosen for the problem. The network will try to minimise this loss function, also known as the empirical risk:

$$\begin{aligned}\theta_{\text{best}} &= \arg\min_\theta \frac{1}{N}\sum_{i=0}^{N-1} L\left(y_i, \hat{y}_i\right) \\ &= \arg\min_\theta \frac{1}{N}\sum_{i=0}^{N-1} L\left(y_i, o_\theta\left(\mathrm{x}_i\right)\right)\end{aligned} \tag{2.4}$$

Multiple optimisation methods can achieve the minimisation of eq. (2.4). The most commonly used method is gradient descent. Gradient descent calculates the gradients for the chosen loss function concerning the model parameters θ. These gradients can be computed by a method called backpropagation [34], which is based on the chain rule of derivatives. These gradients give the direction of change, and the parameters are adjusted by a learning rate γ.

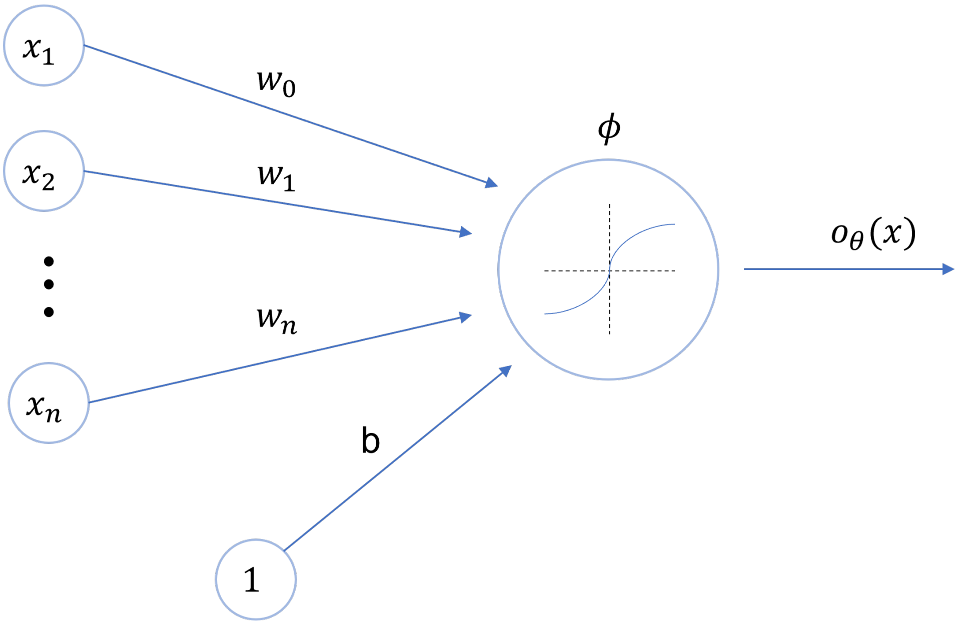


Figure 2.2: Graphical representation of an artificial neuron. The figure shows the inputs x, the weights w, the bias b and an activation function ϕ applied, to produce an outcome $o_\theta(x)$.

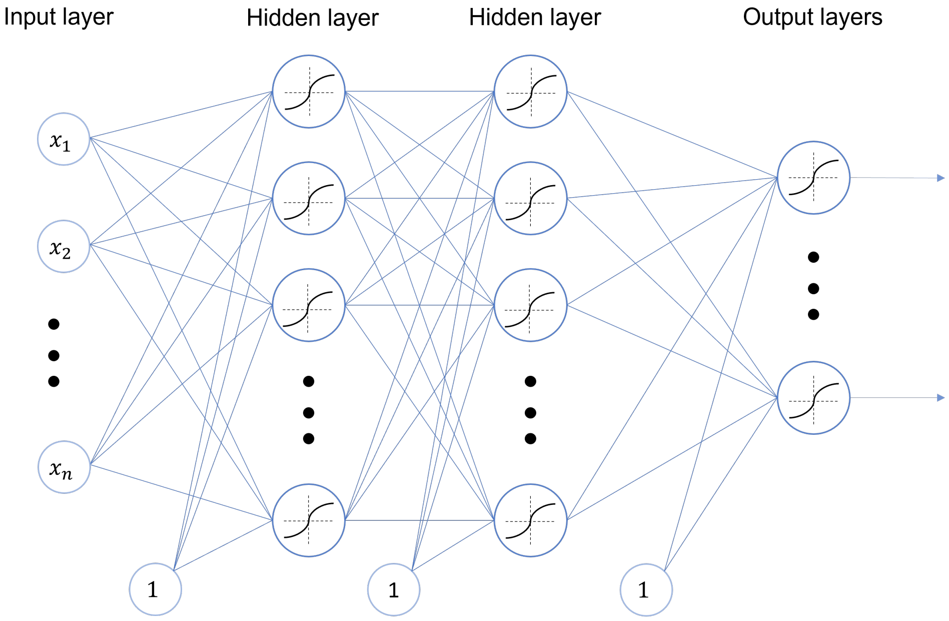


Figure 2.3: Graphical representation of a MLP with multiple hidden layers. The input x is fed through the model's hidden layers, and some bias is added to produce an output.

The multilayer perceptron (MLP) is introduced to enable the mapping of more complex non-linear mappings. The MLP combines multiple neurons into a network. Neurons are stacked to form layers of multiple neurons; a graphical representation is given in fig. 2.3.

It must be noted that a single fully connected layer can achieve the same level of complexity using sufficient neurons. However, the number of parameters to achieve similar complexity is significantly higher. The same principle of backpropagation is used to update the parameters for the MLP.

2.2 Forecasting time series using deep learning models

Lately, there has been an increase in research towards deep learning-based frameworks to forecast temporal data. This increase can also be seen in the field of oceanography. However, forecasting time series is generally considered harder to solve than pattern recognition problems [35]. There are various methods to forecast or improve forecasting using deep models. As a starting point, one could increase the accuracy of current numerical models. In Londhe and Panchang [9] NN are used to forecast the numerical error. However, such hybrid methods are still based on numerical models, so a numerical model must be run for every new forecast.

2.2.1 Forecasting the significant wave height using deep learning models

Forecasting time series is a common problem in all industries, which many authors have tried to solve by employing neural architectures. Oceanography's first efforts using deep models are based on using NNs. Deo et al. [8] published one of the first papers using NNs to produce SWH forecasts. A sequence of wave data is fed into the input layer to produce multiple outputs. With increasing computational power and optimisation techniques, NNs are a frequently used method to produce SWH forecasts [8, 14, 15, 9, 16, 17]. All papers suggest that the NNs have sufficient accuracy, are highly fault-tolerant and can deal with the complexity of the underlying physics.

In these studies, they tend to conserve some temporal dependencies while using NNs, i.a. by varying the number of neurons per input. However, NNs are not known for their capability of interpreting sequential data. A NN might save sequential information in its weights; however, this cannot be controlled. To save sequential dependencies, recurrent models were introduced, e.g. recurrent neural network (RNN), gated recurrent unit (GRU) and long short term memory (LSTM). Especially LSTMs are frequently seen in the metocean literature [18, 19, 20]. LSTMs outperform the other models such as NNs, extreme learning machines (ELMs), support vector machines (SVMs), residual neural network (ResNet) and random forest (RF). The improvement can especially be seen for higher lead times, which could be explained by the memory mechanism available in these cells.

The accuracy of deep models is highly dependent on their architecture and is built by trial and error, educated guesses, empirical arguments and data characteristics. Furthermore, proven model builds are for specific datasets and give no guarantee of effectiveness on other datasets. Therefore it is important to consider multiple architectures. Pushpam P. and Enigo V.S. [19] compares three different LSTM architectures. However, it must be noted that the exploration is limited to a single architecture of vanilla, stacked and bidirectional model.

Papers mostly focus on univariate models to forecast the SWH. However, there is a variety of geographical and meteorological data available which affects the SWH [36]. Li and Liu [21] leverage these variables by causal modelling [37]. A dynamic Bayesian network (DBN) is proposed, a method of causal modelling over multiple time steps [38]. The structure learned can be seen in fig. 2.4.

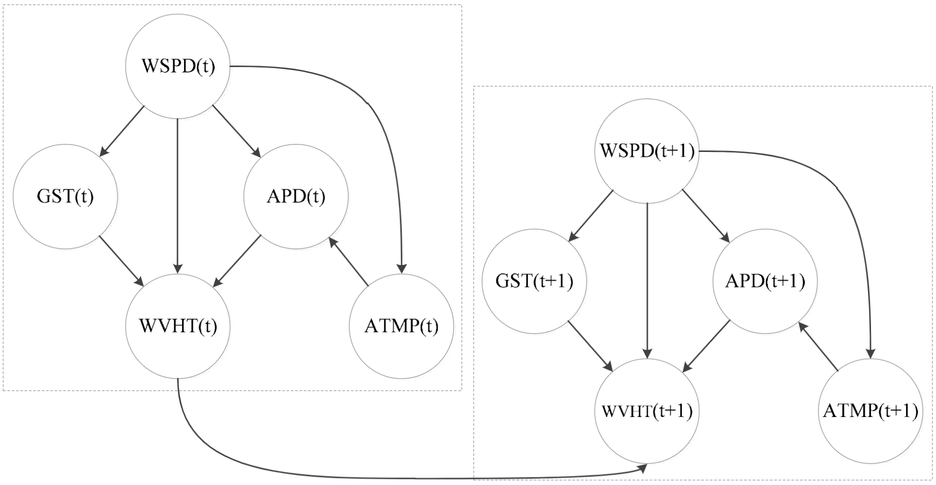


Figure 2.4: The DBN structure to forecast the SWH (referred to as WVHT). Other parameters depicted in this figure are the wind speed (WSPD), the gust speed (GST), the average wave period (APD) and the air temperature (ATMP). The figure is adapted from [38].

The proposed method proves to give better results compared to NNs, SVMs and RF, achieving similar results as papers using recurrent deep models.

2.2.2 Transfer learning for deep recurrent models

Handling machine learning models with limited data is a well-researched topic in the literature. Different techniques and models have been developed to overcome this problem. Known solutions in literature are, e.g. few-shot learning, augmentation, and transfer learning [39]. Transfer learning leverages data from other datasets to increase performance. This seems suitable for the problem at hand, since we can leverage nearby buoys with sufficient data.

Transfer learning, or domain adaptation, is a technique where the model is trained on a source distribution to enhance the accuracy of a target distribution [40]. In fig. 2.5 a schematic representation is given. Transfer learning is a frequently used technique for computer vision and Natural language processing (NLP) tasks [41]. However, there are some essential differences between spatial, NLP and temporal datasets. Deep models learn hierarchically; the first layers will learn more general patterns, e.g. shapes of objects, whereas deeper layers will learn more specific features, e.g. small curves. These hierarchical insights are not available for deep temporal models.

The lack of clear hierarchical representation in time series is an important factor in the transferability of a dataset. If the source and target domains are not interpreted similarly by a NN, it could lead to negative transfer, meaning the accuracy will drop. Research on the transferability of spatial data is still novel and even more sparse for temporal data [43, 22]. In Fawaz et al. [42], Wen and Keyes [44], Ye and Dai [23] experiments are run for time series classification using convolutional neural networkss (CNNs). However, the results revealed no evident conclusion on transfer learning for time series. Despite this, a transfer of weights could potentially significantly increase the accuracy of given models [45, 46]. In these papers, a model is pre-trained on a large dataset, and the fully connected layer is transferred and fine-tuned on the target data. In figure table 5.1, the results are

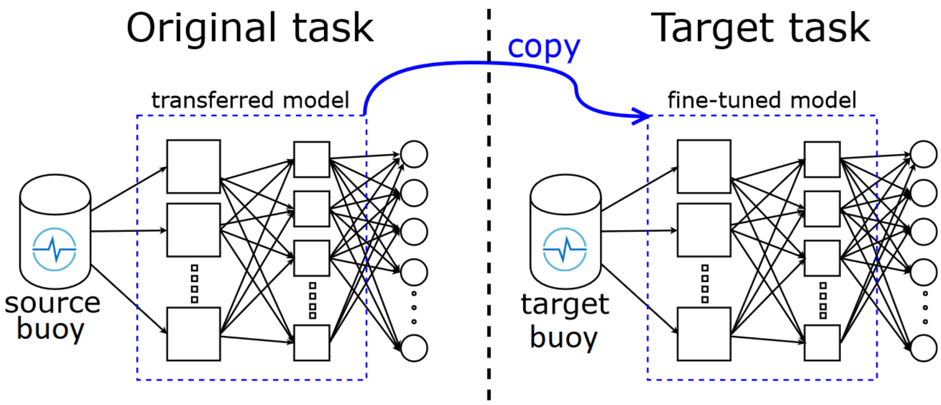


Figure 2.5: A schematic representation of transfer learning process for deep models, applied to buoy data. The weights of a model trained on a source buoy are transferred to forecast at a target buoy in a different location. The figure is adapted from [42].

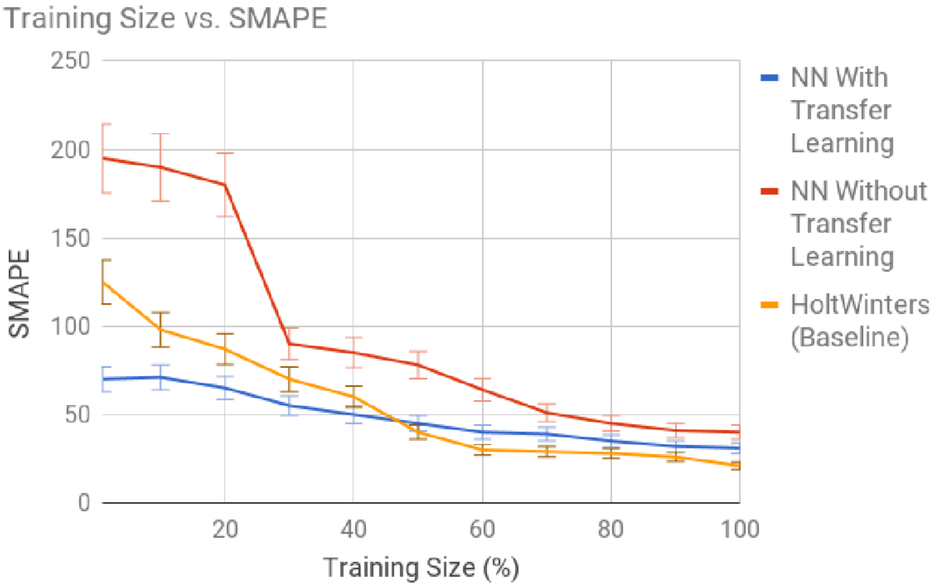


Figure 2.6: Performance comparison of LSTM models between training with single time series and training with transfer learning. The x-axis is the training size; the y-axis is symmetric mean absolute percentage error (sMAPE), where a lower sMAPE indicates better performance. The red curve (top at the left side) represents a time series-trained NN, the blue curve (bottom at the left side) represents a NN using transfer learning and the orange curve (middle at the left side) represents the HoltWinters' baseline used. The performance gap is huge for short training sizes. When training size increases, the performance difference shrinks. The error bar illustrates the standard deviation of sMAPE over 58,000 customers. The figure is adapted from [45].

shown for transfer learning applied on a LSTM framework to forecast residential scale electricity loads at hourly granularity from Pacific Gas and Electric Company. Especially for smaller training sizes, a significant improvement can be seen.

2.2.3 Signal decomposition for optimising model performance

Model accuracy can be improved by increasing the interpretability of the original signal. This can be done by pre-analysing the signal and feeding these findings into the model. Well-known methods for analysing signals tools are Fourier [47] and wavelet [48] decompositions. A Fourier transformation will tell what frequencies are available in the signal. A wavelet transform will tell what frequencies are available where. By locating the frequencies, wavelets can be used to increase performance [49, 50, 51]. However, wavelets assume stationarity of the signal, which is often not true.

Empirical mode decomposition (EMD), a technique first introduced by Huang et al. [52], is able to analyse non-stationary signals. EMD decomposes (or sifts) the signal into multiple Intrinsic mode functions (IMFs) and a residual. These IMFs are a complete set of basis functions, whose amplitude and frequency may vary over time (non-stationarity) [53]. The two most common variations to this algorithm are; ensemble empirical mode decomposition (EEMD), which adds white noise to the initial signal [54], and complete ensemble empirical mode decomposition (CEEMD), which adaptively adds white noise for every sifting iteration [55].

In literature, various research fields introduce EMD as a hybrid combination to enhance forecasting. Such as the metro passengers, the tourist industry, wind speed, ground temperatures, including the SWH [24, 25, 26, 27, 28, 29, 30]. These papers' results significantly outperformed the baseline set, especially for longer forecasting horizons. All papers use a similar framework shown in fig. 2.7.

However, none of these papers considers the empirical nature of EMD. All authors decompose the signal before splitting the dataset into train and test (or validation) sets. The sifting process is based on splines between the maxima and the minima of the whole signal. Details about this process can be found in section 6.1. Since the dataset is not split before EMD, the maxima and the minima of the testing set will be used to determine each IMF. So information from the test set is leaked to the training set via EMD, introducing bias in the model.

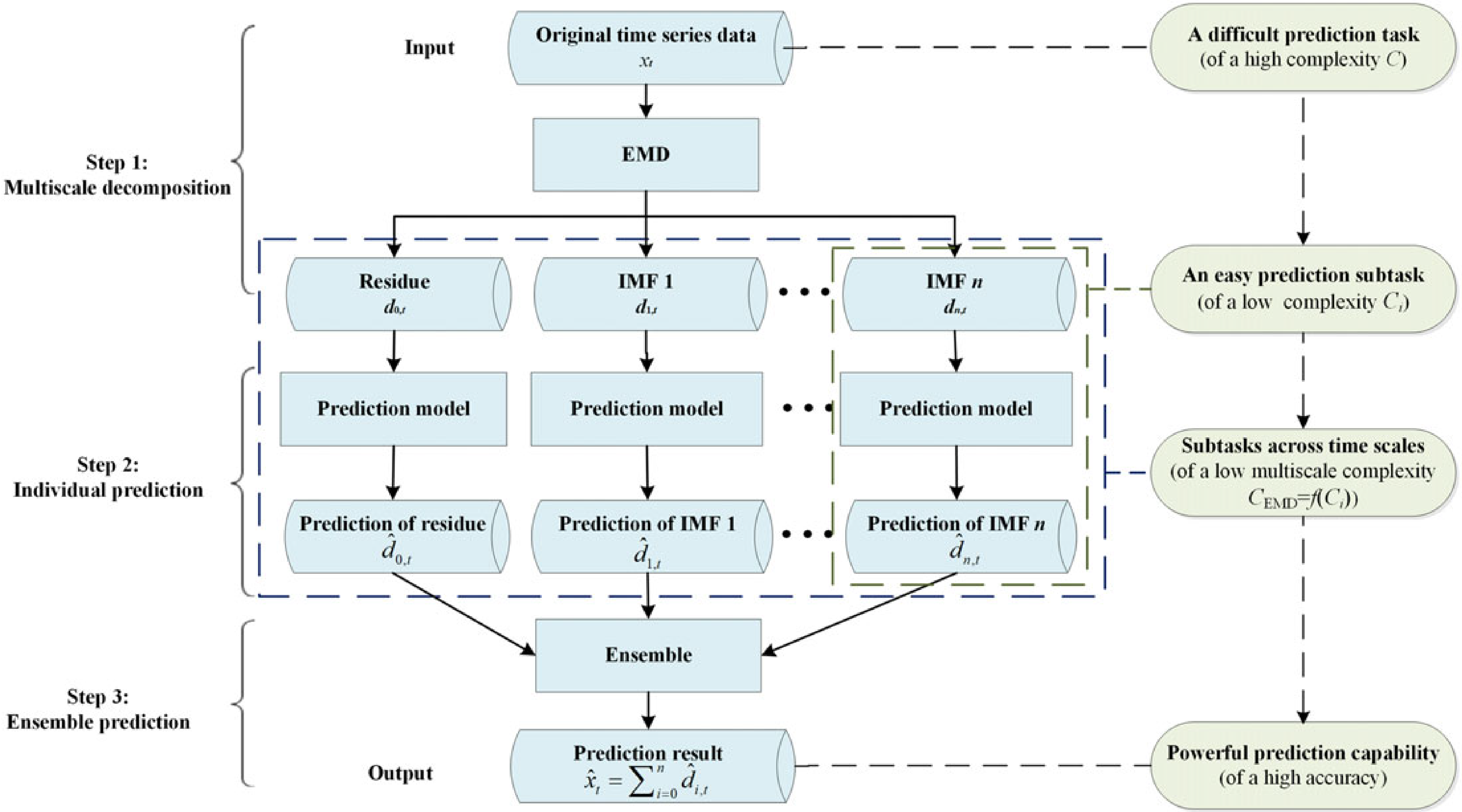


Figure 2.7: Framework of a typical EMD-based ensemble method for forecasting time series data. A signal x_t is decomposed via EMD into n IMFs $d_{n,x}$ and a residual $d_{0,x}$. Each decomposed signal is forecasted individually to produce $\hat{d}$. These forecasts are summed to gain the final forecast of the signal $\hat{x}_t$. The figure is adapted from [56].

Chapter 3

Problem exposition

This chapter gives more context to put this research in a broader perspective and give insight into the decision for the research direction. This thesis is performed in collaboration with Fugro, their interest in this research will be covered first in section 3.1. Then the available data will be discussed in more depth in section 3.2. Finally, a formalisation is made in section 3.3, to clarify the problem.

3.1 Business context

Fugro is a global company founded in the Netherlands that specialises in collecting and analysing geographical data. Fugro tries to gather data-driven insights to enhance safety in diverse geographical fields. They are predominantly active in the infrastructure and energy industries for onshore and offshore projects. Fugro provides offshore weather forecasts for these industries. These forecasts enable companies to take informed decisions and plan efficiently for offshore operations.

Fugro uses oceanographers and meteorologists with metocean models to provide accurate site-specific forecasts. These models are run four times a day and include wind, temperature and wave parameters for any global location. The forecasts are tailored to specific locations and are adjusted and tested with locally measured inputs. To acquire and assimilate data, Fugro uses a variety of sources:

- Data buoys, terrestrial radar and recorded observations
- High-resolution global atmospheric and wave model data from several global modelling suites and in-house models
- Real-time high-resolution geostationary and polar-orbiting satellite imagery
- Synoptic upper-level actual and Global Telecommunications System (GTS) bulletins

As mentioned in section 1.1, these numerical models are computationally expensive, which carries an associated cost for every run. Furthermore, they compute the significant wave height (SWH) over a large grid and are not site-specific. Machine learning techniques could overcome these disadvantages. These only require computational resources during training but are computationally cheap when in use. This would eliminate the execution costs and allow the model to execute at any time. Moreover, these models can forecast specific locations using buoy data.

Fugro is interested in producing forecasts at any given (offshore) location. However, most operations occur in shallow waters (up to 500 meters). Therefore, Fugro slightly prefers forecasting wave heights with a water depth of at most 500 meters, especially since these shallow waters are hard to predict for the numerical models due to increased seabed friction over the water column. However, it must be noted that this thesis does not specifically focus on shallow water.

3.2 Data understanding

As mentioned in section 3.1 Fugro has various ways of obtaining their data. However, these datasets are not publicly available and are bound to confidentiality. In this thesis, the use of data is restricted to publicly available data by the National Buoy Data Center (NBDC) [57], located in the United States. We considered solely data which is acquired via in-situ measurements from buoys. For this study, we focus on wave buoys since these in-situ devices could be deployed at a site of interest. This has the advantage of getting specified geographical data at the site of interest. Furthermore, these in-situ measurements provide accurate local measurements, in contrast to, e.g. satellite measurements over several square kilometres.

The NBDC provides the following information regarding the buoys; buoy specific, meteorological, wave-specific and oceanic data. These are given in the appendix, respectively in tables A.1 to A.4. The equipment can directly measure most features with a given accuracy. For the equipment accuracy, we would like to refer to the site of the NBDC [57]. However, some features are calculated based on measurements and require some assumptions. In this thesis, we will solely discuss the calculation of the SWH; for other features, we would like to refer to the NBDC site [57]. As discussed in section 2.1.1 in eq. (2.2) a summation of the frequency bandwidths is necessary to calculate the SWH. NBDC systems sum over a range of 0.0325 to 0.485 Hz and use an interval of length from 0.005 to 0.02 Hz for the frequency bandwidths.

The NBDC maintains many measurement stations publicly available. However, the availability of data strongly varies per buoy location. Some stations have a lot of missing data and do not measure specific parameters. Therefore it is a necessity to make a selection of buoys to use. The scarcity of the data in the NDBC dataset is closely similar to the problem in which we are interested in this thesis: Enhancing a SWH forecast.

3.3 Formalisation of the problem

Let us define a set of measured sequences

$$\mathbf{X}^b = \{X_1^b \ldots, X_M^b\}, \tag{3.1}$$

where b is the specified buoy and M is the number of measured variables. The variables X^b represent all measurements at buoy b with

$$X_i^b \in \mathbb{R}^{|T_1|} \quad \text{for all} \quad i \in \{1, \ldots, M\}, \tag{3.2}$$

where T_1 is the set of timestamps. Furthermore, let us define Y as the SWH with

$$Y^b \in \mathbb{R}^{|T_2|}, \tag{3.3}$$

where T_2 is the set of time steps following T_1.

Now let us consider buoy k where $|T_1| = 4382$, corresponding to half a year of hourly data points. A model f with parameters ψ_k is used to generate a forecast for Y^k as

$$\hat{Y}^k = f_{\psi^k}\left(\mathbf{X}^k\right). \tag{3.4}$$

This thesis focuses on improving the forecast generated by the model. Two different approaches are explored; by leveraging model parameters obtained from a different training set (ψ^b), to produce $\hat{Y}^k$. And by decomposing the SWH into multiple intrinsic mode functions (IMFs), and producing a model to forecast each IMF individually.

Chapter 4

Forecasting the significant wave height using deep recurrent networks

To apply techniques to improve the significant wave height (SWH) forecast, we first have to set a baseline for the deep models to improve. Therefore, we start with analysing models using sufficient data. In section 4.1, we discuss the theory necessary to understand the models and their usage. Then the available datasets are analysed and discussed in section 4.2. The data analysis is used to create the experimental setup in section 4.3. Finally, the results will be shown in section 4.4, which will be interpreted in section 4.5.

4.1 Theory for forecasting models

This section will discuss the necessary theory relevant to this chapter. First, deep recurrent networks are discussed. Then the autoregressive (AR) model is discussed, which is used as an extra baseline. Finally, the accuracy metrics used to evaluate all the models are discussed.

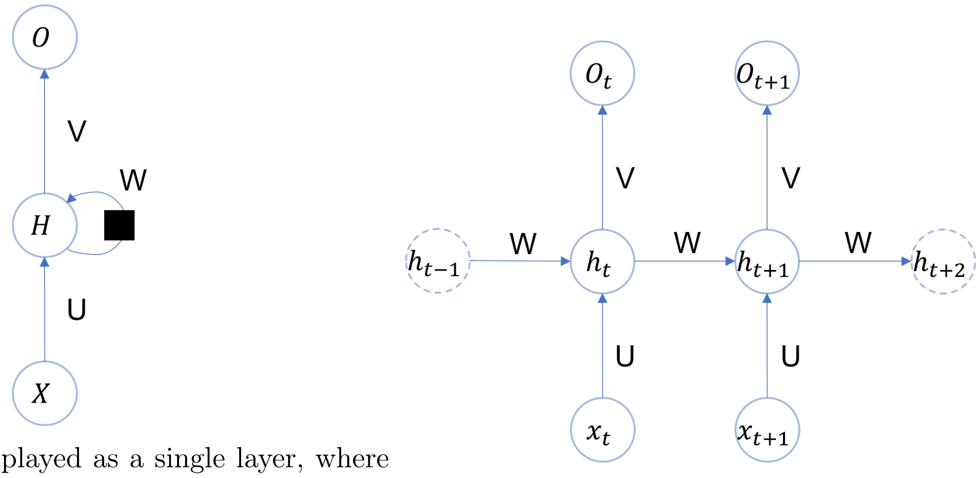
4.1.1 Recurrent neural network

Neural networks (NNs) are based on the assumptions that the input data are independent samples. However, this does not hold for time series. Therefore a variation of the NN is introduced; the recurrent neural network (RNN), designed to exploit sequential information. An RNN processes the input orderly and stores memory of previous elements. RNNs try to select and retain relevant information, allowing them to capture temporal dependencies.

An RNN cell receives two inputs, one external and one via a feedback connection. For each recursion in a cell, the cell receives a new input value x_t and the hidden state h_{t-1} from the previous time step. Then the hidden state of the cell is updated, and finally, an output o_t is given. For this process, the RNN uses three different weight matrices, and the outputs can be calculated according to

$$\begin{aligned} h_t &= \phi_1\left(Ux_t + Wh_{t-1} + b_h\right) \\ o_t &= \phi_2\left(Vh_t + b_o\right). \end{aligned} \tag{4.1}$$

Where U represents the weights between the input and hidden layer, W are the weights for the feedback loop between the two hidden states, and b_h is a bias term. To generate



(a) An RNN displayed as a single layer, where the black square indicates a delay.

(b) An RNN displayed unrolled.

Figure 4.1: A graphical representation of an RNN cell. With X the input matrix, H the hidden state matrix, O the output matrix and U, W, and V the weight matrices. Figure (b) shows a rolled-out version of a similar network. Note that the input and the hidden state are lowercase for figure (b), which concerns single vectors. Furthermore, the bias term to compute the hidden state and the output are not shown in this figure.

the output, a weight matrix V is used with a bias term b_o. Furthermore ϕ_1 and ϕ_2 are the activation functions. A graphical representation of the flow of information through an RNN is given in fig. 4.1.

The gradients for an RNN are, similarly to a NN, computed via backpropagation. However, the recursive nature of an RNN leads to exploding or vanishing gradients [58]. This compromises the ability to learn long-term dependencies.

Long short term memory

To resolve exploding and vanishing gradients a long short term memory (LSTM) cell is introduced [59]. These cells use a memory cell with a gating mechanism, which protects the state of the RNN H. This memory cell decides what, when and how to process information. This allows the model to train the parameters of the cell efficiently, especially for long-term dependencies.

In the LSTM memory cell C is introduced for the gating mechanism, where the information can be controlled based on the hidden state and the input. The LSTM cell uses a memory cell in combination with three gates; a forget gate, an add gate and an output gate. These gates multiply the cell values with a weight vector W. This vector consists of zeros and ones and is therefore able to "remember" and "forget" values. The values of this vector are specified by the input x_t and the hidden state h_{t-1}:

$$f_t = \sigma\left(W_f\left[h_{t-1}, x_t\right] + b_f\right), \tag{4.2}$$

where σ is a logistic sigmoid function. The add gate creates a vector of candidate values C'_t, based on the input x_t and the hidden state h_{t-1}

$$C'_t = \tanh\left(W_C\left[h_{t-1}, x_t\right] + b_C\right). \tag{4.3}$$

These candidate values are then processed by the add gate, where the decision is made

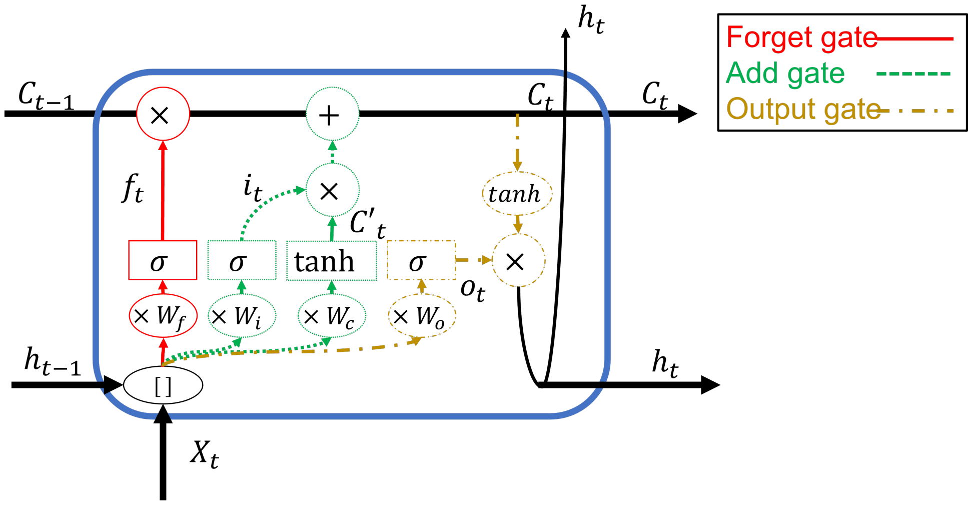


Figure 4.2: Schematic representation of the LSTM cell and its gating mechanism. The cell receives an input X_t, the hidden state h_{t-1} and the C_{t-1} to process. These inputs are processed via the gate's weights W and an activation functions σ and tanh to produce the outcomes for the next recurrence.

whether to add a value or not, similar to the forget gate;

$$i_t = \sigma\left(W_i\left[h_{t-1}, x_t\right] + b_i\right). \tag{4.4}$$

The forget and the add gate together produce the new cell state by

$$C_t = f_t \cdot C_{t-1} + i_t \cdot C'_t. \tag{4.5}$$

This new cell state is again gated by the values of the input x_t and the hidden state h_{t-1};

$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right). \tag{4.6}$$

Where finally, the output is produced by

$$h_t = o_t \cdot \tanh\left(C_t\right). \tag{4.7}$$

A schematic overview of this process within an LSTM cell is given in fig. 4.2.

Gated recurrent unit

A popular variation to the LSTM cell is the gated recurrent unit (GRU), which also solves the gradient problem by gating [60]. The GRU uses two gates; an update gate z and a reset gate r. The update gate preserves what information should be updated, and the reset gate determines what should be ignored from the previous hidden state. These gates have two weight vectors W and U, one for the hidden state and one for the input. The update and the reset gate can be computed as

$$\begin{aligned} z_t &= \sigma\left(W_u h_{t-1} + U_u x_t + b_u\right), \\ r_t &= \sigma\left(W_r h_{t-1} + U_r x_t + b_r\right). \end{aligned} \tag{4.8}$$

Then the update as proposed by the cell is computed as

$$\tilde{h}_t = \tanh\left(W_c\left(r_t \cdot h_{t-1}\right) + U_c x_t + b_c\right). \tag{4.9}$$

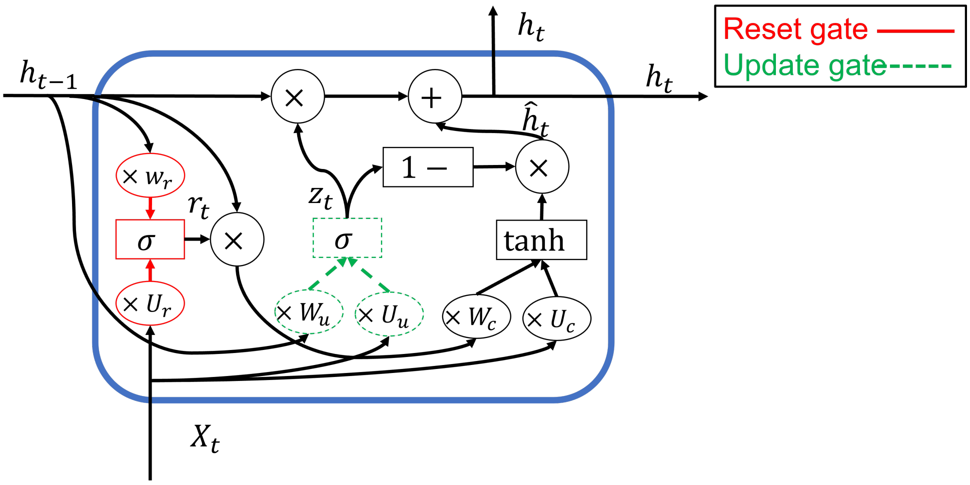


Figure 4.3: Schematic representation of the GRU cell and its gating mechanism. The cell receives an input X_t and the hidden state h_{t-1} to process. These inputs are processed through the gate's weights W via activation functions σ and tanh to produce the outcomes for the next recurrence.

With this update, the new value for the hidden state can be computed as

$$h_t = z_t \cdot h_{t-1} + (1 - z_t) \cdot \tilde{h}_t. \tag{4.10}$$

A schematic overview of this process within an GRU cell is given in fig. 4.3.

Recurrent Architectures

Recurrent cells can be stacked into multiple layers L, where the input of L^{th} layers x_t equals the previous layers' hidden state. Next to that, the recurrent network can be bidirectional, where the data is processed in both directions [61]. The network can simultaneously process future and past elements by encoding the sequence in the opposite direction. The output depends on two hidden states; the forward h_t^f and the backward h_t^b. In fig. 4.4 a stacked bidirectional RNN is given, where the model creates output vector O_t for every time step t. Note that both LSTM and GRU cells can be fit in this example. Regardless of the architecture used, the output produced by the recurrent cell requires some interpretation to obtain a (output) vector of the desired length, typically done using a NN.

4.1.2 Autoregressive model

An AR model builds on the assumption that the current observation is correlated to a weighted sum of the past p values. The model tries to find weights φ_i and a constant c. The AR model is defined as

$$X_t = c + \sum_{i=1}^{p} \varphi_i X_{t-i} + \varepsilon_t, \tag{4.11}$$

with some white noise ϵ_t.

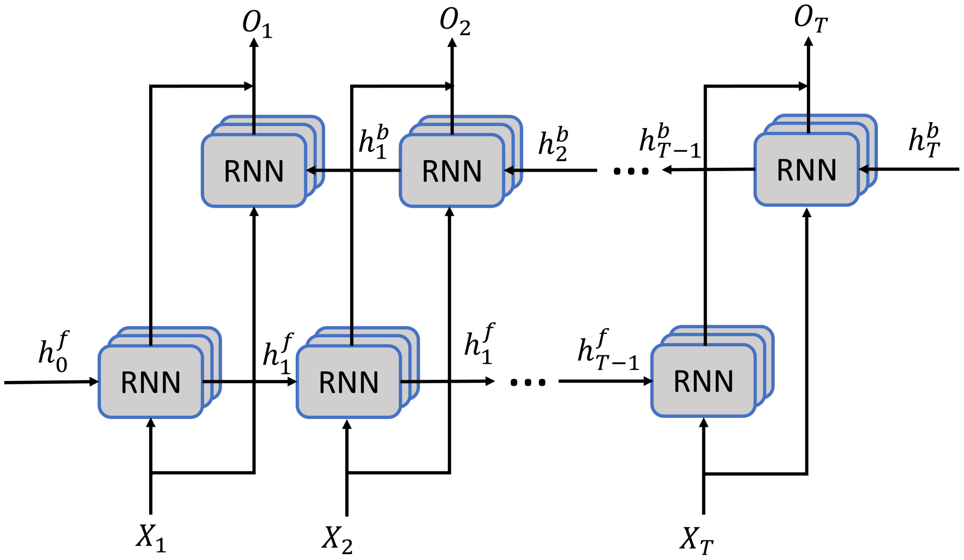


Figure 4.4: Schematic representation of a three-layer stacked bidirectional RNN. The superscript of the hidden states depicts the direction of information flow. The input is depicted as X and the output as O.

4.1.3 Accuracy metrics

Three accuracy metrics quantify the difference between the actual value y_i and the forecasted value $\hat{y}_i$. The root mean squared error (RMSE) is defined as

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}. \tag{4.12}$$

The symmetric mean absolute percentage error (sMAPE) is defined as

$$\mathrm{sMAPE} = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{(|y_i| + |\hat{y}_i|)/2}. \tag{4.13}$$

And the Pearson correlation (PCC) is defined by

$$\mathrm{PCC} = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(\hat{y}_i - \bar{\hat{y}})}{\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2 \cdot \sum_{i=1}^{n}(\hat{y}_i - \bar{\hat{y}})^2}}, \tag{4.14}$$

where the bar indicates the sample mean.

4.2 Exploratory data analysis

Two buoys were selected east of the Bahamas in the Northern Atlantic Ocean. These buoys are relatively close (508 km) and can be considered in the same climate. Furthermore, these two buoys have sufficient data to capture the variability caused by climate patterns such as El Niño and La Niña. The buoys provide measurements at an hourly resolution for 2017 until 2020. This exploratory analysis will focus on the SWH and the correlation with other parameters. Further measurement characteristics and basic

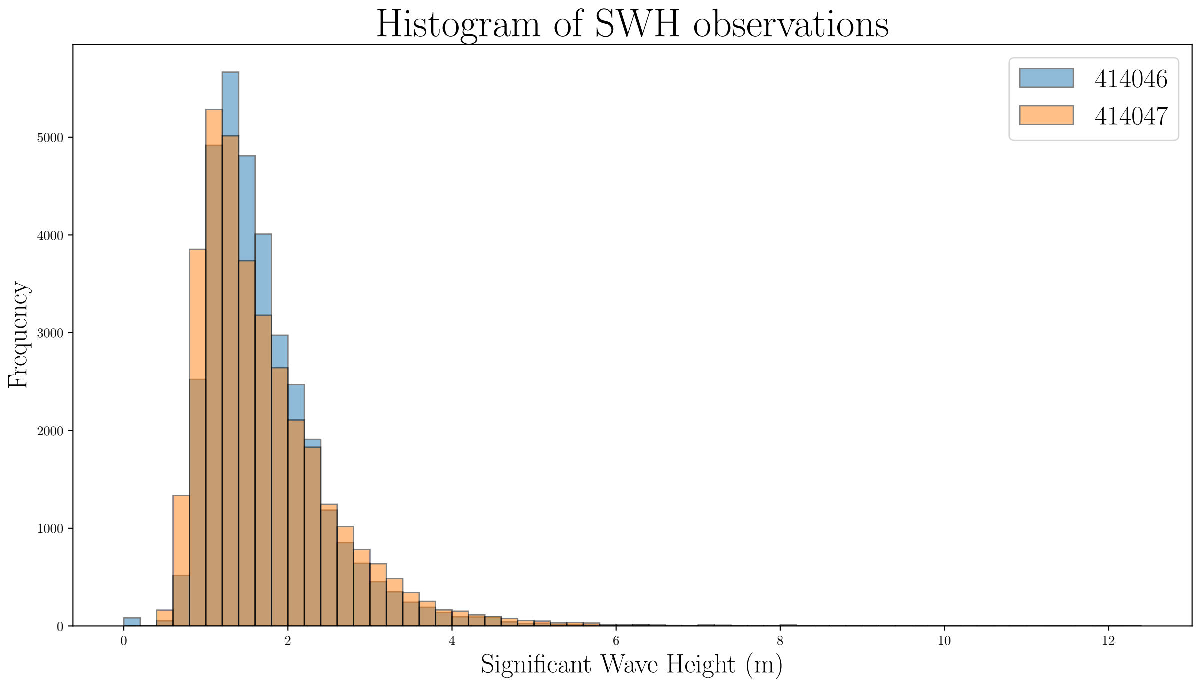


Figure 4.5: A histogram for all SWH observations for buoys 414046 and 414046. The histogram uses a bin size of 0.1.

information, for respectively buoy 414046 and 414047, can be found in the appendix in tables B.1 and B.2. However, it is notable that the basic characteristics of both buoy measurements are similar.

An overview of the SWH observations and their missing values are given in table 4.1. The proportion of missing values is not significant. However, most missing values are in consecutive order due to maintenance or equipment failure. The monthly distribution for the missing values can be seen in the appendix in fig. B.1. Salient missing values are for buoy 414046 in May 2019 since no data is available from 29 April 2019 until 08 May 2019.

Buoy	No. of observations	No. of missing values	Percentage missing values
414046	34 427	637	0.018 %
414047	34 826	238	0.007 %

Table 4.1: The number of available SWH observations for buoy 414046 & 414047 and their missing values.

The distribution of available observations for both buoys is shown in fig. 4.5. Both distributions are skewed to the right and have a positive kurtosis. A Shapiro-Wilk test confirms the rejection of normality (appendix table B.3). Box-Cox transformations could increase normality, resulting in variance stabilisation, increased stationary behaviour, and better performance of some classical time series models [62]. However, normality could not be achieved by applying a Box-Cox transformation (appendix fig. B.2 and table B.3).

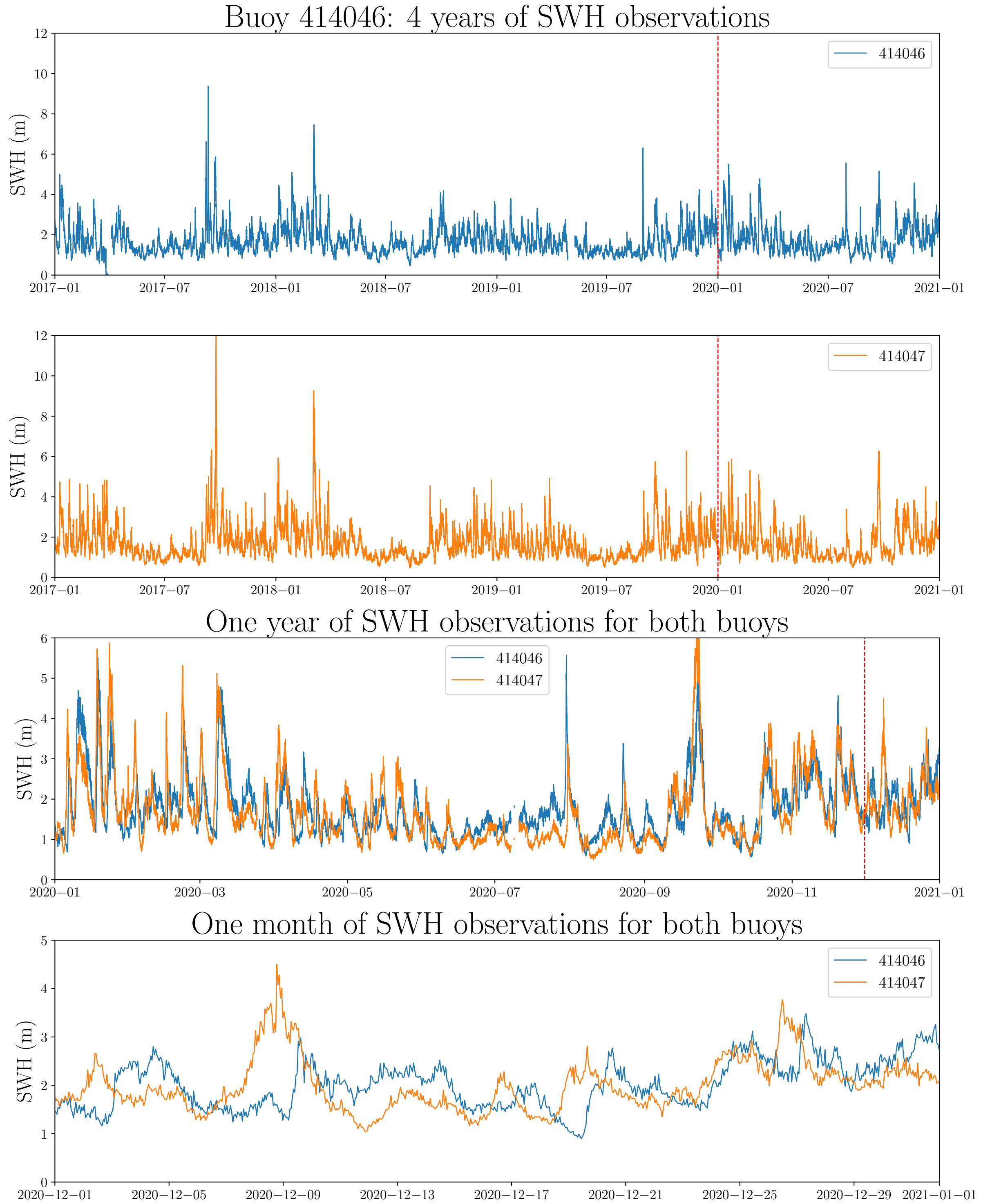
A visual representation over 2017 until 2020 for the SWH can be seen in fig. 4.6. The SWH experience similar temporal wave patterns at the two locations. Considering the year and month plot, we can see that the SWH are correlated. Peak values can be linked to local hurricanes, e.g. September 7, 2017 - Hurricane Irma and September 1, 2019 - Hurricane Dorian. Furthermore, we can see some yearly seasonal variations; the SWH near July seems to have relatively low waves. Considering the whole signal, it seems stationary, with no systematic changes in mean and variance. Whether the signal is stationary is checked by performing an augmented Dickey-Fuller (ADF) test, using a null hypothesis of the signal being non-stationary. The test is executed over linear interpolated data with a significance level of 0.05. The test rejects the null hypothesis, suggesting a stationary signal for both buoys (appendix in table B.4).

For this research, we are interested in forecasting the SWH with a small interval (up to two days). If we consider the monthly plot, the signal seems more erratic. The fluctuation of the variance is greater compared to the yearly plot. Furthermore, we can see shifts in the average, indicating non-stationary behaviour. We can analyse this behaviour by applying the ADF test to smaller time intervals. Changing the interval also influences the ADF since smaller intervals exhibit less reversion to the mean. The null hypotheses for semi-annually and annually intervals are still rejected; however, they cannot be rejected for monthly or weekly intervals. The details of all the ADF tests are shown in the appendix in table B.4.

The focus of this thesis lies in the use of deep models. However, classical time series models could be as effective and require less computational power. Therefore we consider these models as an extra baseline for the deep models. These models are based on a correlation of the signal with their lagged versions. To determine what classical models would suffice for forecasting, we consider the autocorrelation function (ACF) and partial autocorrelation function (PACF) plots given in figs. 4.7 and 4.8. The ACF measures the direct and indirect correlation between lagged versions. The PACF tries to correct the indirect correlation by removing already found relations. Besides indicating what type of time series model would be viable, the correlations are also relevant for the input of deep recurrent models. NNs could be seen as autocorrelation functions since they linearly map an input to an output. The significance of lagged values indicates the importance of sequence length for the input. Finally, ACF and PACF check for randomness and indicate the amount of random noise.

In the ACF, the correlation tails off (geometric decay) with increasing time lag. If the lag increases even further the function will follow a damped sine function (appendix fig. B.3). The ACF suggest that the most recent values affect the current value the most. According to Bartlett's test the function loses its statistical significance after approximately 1100 hours (roughly 1.5 months) for both signals. If we study the PACF, we can see that the residual correlation is negligible. The PACF plot gives correlation for the first two lagged values and is cut off after. Both plots indicate correlations meaning the data does not entirely consist of white noise. Furthermore, we could observe that the most recent observations contain significantly more information about the current time step than larger lags.

The correlation between the SWH and other available measurements is analysed via the PCC. In table 4.2 the three most significant correlations are given. these correlations match the direct causal relations found in Li and Liu [21], seen in fig. 2.4. In the appendix a heat map for all correlations is given in fig. B.4.



Buoy 414046: 4 years of SWH observations

One year of SWH observations for both buoys

One month of SWH observations for both buoys

Figure 4.6: Time series of 4 years of SWH observations for buoy 414046 and 414047. Three different intervals are given; the whole dataset, one year of observations and one month of observations. The dashed line indicates where the cut is made for the zoomed intervals.

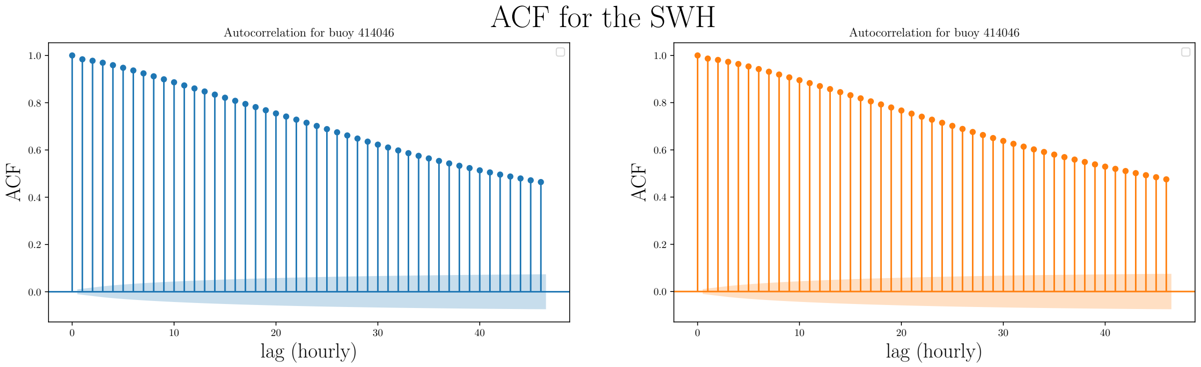


Figure 4.7: The ACF plot for the SWH for buoy 414046 and 414047. The coloured cone depicts the 95 % interval according to Bartlett's test and is an indicator of the significance threshold. Anything outside the cone can be seen as statistical significant.

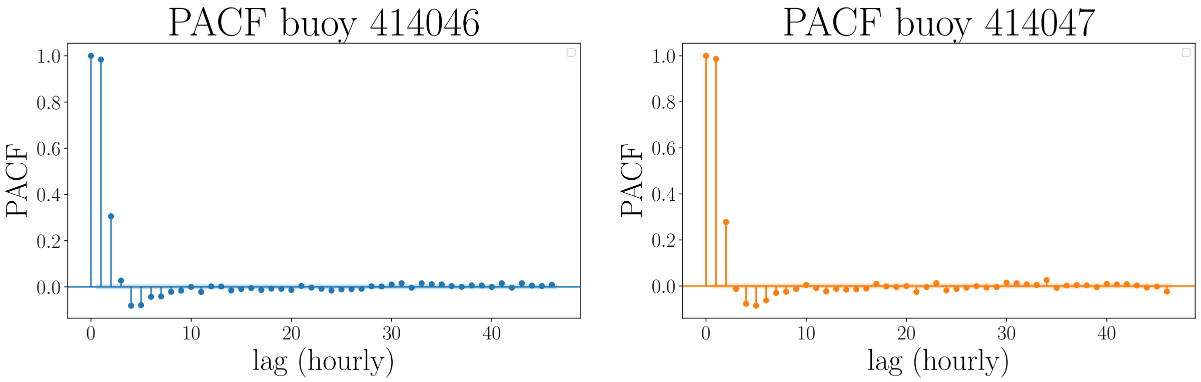


Figure 4.8: The PACF plot for the SWH for buoy 414046 and 414047. The coloured cone depicts the 95 % interval according to Bartlett's test and is an indicator of the significance threshold. Anything outside the cone can be seen as statistical significant.

Buoy	**gust speed (GST)**	**average wave period (APD)**	**wind speed (WSPD)**
414046	0.66	0.65	0.55
414047	0.60	0.64	0.61

Table 4.2: The three most significant correlations between the SWH and available measurements for buoys 414046 and 414047.

4.3 Experimental setup

In this section, the experimental setup is discussed. First, the models used are discussed and then their parameters and input values in section 4.3.1. Then we will discuss the data preparation and all training details in section 4.3.2.

4.3.1 Model architectures

As discussed in section 4.2, a half-year signal seems stationary, the ACF is damped sinus, and the PACF shows two lagged correlations. Therefore suggesting that an AR model would suffice. Since the second lag is already considerably lower, both AR(1) and AR(2) models will be considered. These models are applied to subsets of the whole set containing half a year of measurements (4382 data points), and all the forecasts will be averaged. These subsets are polynomially interpolated, and we ignore subsets with missing values in the last 48 hours.

In this thesis three different types of deep sequential models are used: RNNs, GRUs and LSTMs. As discussed in section 6.1, LSTM and GRU cells are more suitable to capture long term relations. While the mechanics of the cells are quite similar, they tend to outperform each other for different datasets [63]. Similar architectures and training procedures are used for all three models to compare the performance.

The length of the hidden state determines how many ways the model could interpret the input. However, increasing the size of the hidden state does not necessarily increase accuracy and generates extra computational costs. The most common hidden sizes are 32, 64, 128 and 256, which all are used for the task at hand. The output of the recurrent cells is fed to a fully connected layer to generate a singular forecast from the hidden states. However, a single layer could have difficulty interpreting the hidden state with increasing size. Therefore we experiment with multiple fully connected layers. Furthermore, we implement stacking cells ($L \in \{1, 2\}$) and bidirectional training. This results in 40 different variations for each recurrent network. An overview of the basic model architectures can be seen in table 4.3. Note that all these architectures are also built stacked and bidirectional.

Model No.	Hidden size	FC layer 1	FC layer 2	FC layer 3	FC layer 4
1	32	32	-	-	-
2	64	64	-	-	-
3	64	64	32	-	-
4	128	128	-	-	-
5	128	128	64	-	-
6	128	128	64	32	-
7	256	256	-	-	-
8	256	256	128	-	-
9	256	256	128	64	-
10	256	256	128	64	32

Table 4.3: The architectures for the experiments with the deep recurrent models. The hidden size for the RNN, GRU and LSTM and the number of neurons for the fully connected layer are given. The last fully connected layer outputs a singular forecast.

The recurrent deep models can process multivariate datasets. However, sequential models use an iterative process to produce multiple forecasts. This is a restrictive factor for multivariate modelling. Each iteration would need a forecast for each input variable which accumulates uncertainty. Therefore the multivariate models are limited to forecasting one value. We limited the multivariate models to a 3 hour forecast, using training sequences with a step size of three ($\{3k|k \in [0, 16]\}$). Experiments are run with three different inputs, using four variables based on their correlation with the SWH (table 4.2). The training sets will be; $\{\mathrm{SWH}, \mathrm{GST}\}$, $\{\mathrm{SWH}, \mathrm{GST}, \mathrm{APD}\}$ and $\{\mathrm{SWH}, \mathrm{GST}, \mathrm{APD}, \mathrm{WSPD}\}$. We limit the multivariate experiments to the best-performing architectures found by the univariate experiments.

4.3.2 Data preparation and training details

The sequence length of the input is limited by vanishing gradients. A longer input will not harm the accuracy itself; however, longer sequences result in smaller training sets which could harm the accuracy. The ideal sequence length depends on which model, the model parameters and the complexity of the data. Recurrent models could process relatively simple signals (e.g. a regular sinusoïde) or sentences in Natural language processing (NLP) up to 1000 steps [64]. However, the longer the sequence, the more context is lost. For NLP sequential models can distinguish context sharply up to 50 tokens [65].

The underlying physical processes causing wave generation are complex and based on multiple environmental factors. The deep models only use historical data of the SWH and a few other variables to produce a forecast. Since we do not use all physically relevant variables, we lose information. Therefore, the models need to acquire more (complex) information from the given variables to generate a forecast. Therefore, we choose a small sequence length to enable the models to still find complex patterns in the data.

The ACF plots showed that the first two days have the most significant auto-correlation. Furthermore, the PACF showed that smaller time lags have a greater correlation. Therefore, a two-day sequence ($[0, 48]$) is used as a training sequence. A smaller training set is also introduced to prevent losing information by vanishing gradients. This training set covers the last two days, including a step size of three hours ($3k|k \in [0, 16]$). Each sequence in the training sets is labelled with the subsequent values to construct a supervised training set.

During the training of the deep recurrent models, we discard sequences with more than two consecutive values missing. The remaining sequences are polynomial interpolated, note that we do not extrapolate. The data is split into a training (70 %), validation (20 %), and test set (10 %). In section 4.2, we found that the SWH observations are not normally distributed. Therefore we use normalisation to improve convergence and generalisation of the deep recurrent models [66].

Training details deep models

All models are trained on a training set fed in batches of 64. The mean squared error (MSE) is monitored on the validation set during training. Once this value does not increase for 5 epochs (patience of 5), the training is stopped, and the best-performing weights are saved. The models are optimised using an Adam optimiser with $\beta = (0.9, 0.99)$. We experimented with three different learning rates γ 10^{-3}, 10^{-4} and 10^{-5}. The training times per epoch varied from 3 s up to 7 h for the most complex models. The models are run on an AMD Ryzen 9, 5900x 12-Core processor, with 32 GB RAM and a NVIDIA GeForce GTX 1070. In total 720 univariate models for each buoy, resulting in a total of 1440 training runs. Furthermore, 12 experiments are run with multivariate models. All deep learning models are implemented and run using python 3.9.7 using the PyTorch library [67].

Evaluation metrics

Multiple error metrics are used to evaluate and compare model performances. The RMSE is used, which is easy to interpret due to their scale dependency. The RMSE has some bias towards larger errors and is frequently dominated by errors on peak values. Since we are especially interested in forecasting peaks, this property is not restrictive. As a non-dimensional (or percentage metric) the sMAPE is used. Furthermore, the PCC is used to measure the similarity between the forecast and the actual observation. These metrics are all compared to a naive baseline, a persistence algorithm which naively predicts the last seen value. If the AR models outperform the persistence algorithm, the AR models will be set as the new baseline.

Each model is evaluated on average performance per given variable for the two buoys. A summary of the number of training procedures/architectures is given in table 4.4.

Model variable	No. experiments
Model type	3
Input-sequence	2
Stacked cells	2
directional training	2
learning rate	3
Model architecture	10

Table 4.4: Summary for the number of experiments for the deep recurrent univariate models. In total, 720 experiments were run for each dataset 414046 and 414047.

4.4 Results

This section presents the results for forecasting the SWH. First, we will discuss the results of the univariate models. Then, the results of the multivariate models will be discussed.

4.4.1 Univariate forecasting

First, we will discuss the persistence model and the two AR models to set a (naive) baseline to validate the effectiveness of the models. The results for these three models are given in table 4.5. For all models the RMSE, sMAPE and PCC are determined over the testing set, containing data over the period 7 August 2020 till 1 January 2021.

The persistence model outperforms the AR models for the RMSE and PCC, only for a higher forecasting horizon we see some improvement for the AR models. If we consider the sMAPE, the AR model gives a slight improvement. However, we can explain this because the ARs are bounded by their coefficients. When the model encounters outliers, it will move towards its constant. This could also explain the models' difference between

		Persistence			AR(1)			AR(2)		
Buoy	Lead time (h)	RMSE (m)	sMAPE (%)	PCC	RMSE (m)	sMAPE (%)	PCC	RMSE (m)	sMAPE (%)	PCC
414046	1	0.14	5.2	0.98	0.15	5.1	0.98	0.15	4.8	0.98
	3	0.19	6.6	0.97	0.20	6.6	0.96	0.20	6.4	0.96
	6	0.26	9.2	0.93	0.28	8.89	0.90	0.27	8.5	0.92
	12	0.38	13.6	0.86	0.39	12.5	0.83	0.39	12.0	0.84
	24	0.54	20.6	0.71	0.53	17.56	0.67	0.54	17.0	0.67
	48	0.73	28.2	0.48	0.66	22.6	0.39	0.67	22.2	0.42
414047	1	0.14	5.5	0.99	0.17	5.3	0.98	0.17	4.8	0.98
	3	0.18	6.6	0.98	0.23	7.1	0.96	0.23	5.6	0.96
	6	0.26	9.1	0.96	0.32	9.8	0.93	0.32	8.5	0.92
	12	0.38	13.6	0.91	0.45	14.3	0.86	0.45	12.0	0.84
	24	0.57	20.1	0.80	0.62	20.3	0.69	0.64	17.34	0.68
	48	0.84	28.9	0.57	0.80	26.6	0.42	0.83	22.2	0.42

Table 4.5: The accuracy metrics for the persistence, the AR(1) and the AR(2) models.

larger forecasting horizons. Since the models arguably perform similarly, the persistence model will be used as a baseline to evaluate the experiments.

Model types

All results are averaged per variable of interest to analyse the effectiveness of the model variables. In table 4.6 the results of three different recurrent cells are given. The results are relative to the persistence model in such a manner that positive values represent improvement concerning the persistence model. Note that negative zeros are given due to rounding.

The RNN and GRU models produce similar results as the persistence model for buoy 414046. The accuracy for buoy 414047 is worse, especially considering longer lead times. Overall, the LSTM generates better results, while performance increases with increasing lead time.

Input sequence

Now let us consider the difference between the two input sequences; successive and stepped. In table 4.7 the relative metrics are given. Note that the stepped input cannot produce a forecast at the first hour. There is a significant difference between the two datasets for the successive input. However, this difference is not clearly visible for the stepped input. Furthermore, the stepped input models show improvement for both buoys, while the successive show equal or worse accuracies.

Stacking cells

To evaluate the effectiveness of stacking recurrent cells, let us consider the results in table 4.8. Both datasets perform similarly, where the stacked model performs negligibly better than a singular layer.

		RNN			**LSTM**			**GRU**		
Buoy	Lead time (h)	RMSE (m)	sMAPE (%)	PCC	RMSE (m)	sMAPE (%)	PCC	RMSE (m)	sMAPE (%)	PCC
414046	1	0.01	0.2	0.00	-0.00	-0.1	-0.00	0.01	0.3	0.00
	3	-0.01	-0.4	-0.00	-0.01	-0.7	-0.01	-0.00	-0.3	-0.00
	6	0.01	0.1	0.02	0.01	-0.0	0.01	0.01	0.3	0.01
	12	0.03	0.4	0.03	0.04	1.3	0.03	0.03	0.7	0.03
	24	0.03	0.3	0.09	0.09	3.8	0.08	0.03	0.5	0.08
	48	0.02	-1.6	0.14	0.13	5.5	0.15	0.01	-2.1	0.12
414047	1	-0.02	-0.7	-0.02	-0.00	0.2	-0.00	0.00	0.4	-0.00
	3	-0.03	-1.7	-0.02	-0.02	-1.1	-0.00	-0.01	-0.6	-0.00
	6	-0.02	-2.0	-0.00	0.00	-0.9	0.00	-0.00	-0.8	0.01
	12	-0.03	-4.1	0.01	0.03	0.1	0.02	-0.01	-2.4	0.03
	24	-0.12	-10.9	0.05	0.10	2.2	0.06	-0.05	-7.8	0.07
	48	-0.37	-23.7	0.14	0.18	4.8	0.15	-0.20	-20.6	0.14

Table 4.6: The average accuracy metrics relative to the persistence model, for the RNN, LSTM and GRU models. Note that positive values indicate improved performance.

		Successive $\{0, \ldots, 48\}$			**Step** $\{3k \mid k \in [0, 16]\}$		
Buoy	Lead time (h)	RMSE (m)	sMAPE (%)	PCC	RMSE (m)	sMAPE (%)	PCC
414046	1	0.0	0.1	0.0	-	-	-
	3	-0.0	-0.2	-0.0	-0.01	-0.7	-0.01
	6	-0.0	-0.3	0.0	0.03	0.5	0.02
	12	-0.0	-0.4	-0.0	0.08	2.0	0.07
	24	-0.0	-0.3	0.0	0.11	3.3	0.17
	48	-0.0	-1.0	0.0	0.11	2.2	0.27
414047	1	0.07	2.7	-0.50	-	-	-
	3	-0.02	-1.0	-0.01	-0.02	-1.2	-0.01
	6	-0.03	-1.7	-0.01	0.02	-0.8	0.01
	12	-0.07	-3.7	-0.01	0.06	-0.6	0.04
	24	-0.15	-8.8	-0.00	0.11	-2.2	0.12
	48	-0.39	-18.5	0.00	0.12	-7.8	0.28

Table 4.7: The average accuracy metrics for the deep recurrent models using different input sequences. Note that positive values indicate improved performance.

		Single recurrent cell			**Stacked cell**		
		$L = 1$			$L = 2$		
Buoy	Lead time (h)	RMSE (m)	sMAPE (%)	PCC	RMSE (m)	sMAPE (%)	PCC
414046	1	0.00	0.1	0.00	0.00	0.2	0.00
	3	-0.01	-0.5	-0.01	-0.01	-0.4	-0.01
	6	0.01	0.1	0.01	0.01	0.2	0.01
	12	0.03	0.7	0.03	0.04	0.9	0.03
	24	0.05	1.4	0.09	0.05	1.6	0.08
	48	0.05	0.6	0.14	0.06	0.7	0.14
414047	1	-0.01	-0.2	-0.01	-0.00	0.2	-0.01
	3	-0.02	-1.3	-0.01	-0.02	-1.0	-0.01
	6	-0.01	-1.4	-0.00	-0.00	-1.1	0.01
	12	-0.00	-2.3	0.02	-0.00	-2.0	0.02
	24	-0.02	-5.8	0.06	-0.02	-5.2	0.06
	48	-0.13	-13.7	0.14	-0.13	-12.6	0.14

Table 4.8: The average accuracy metrics for the deep recurrent models using single and stacked cells. Note that positive values indicate improved performance.

Uni- and Bidirectional training

Similar results are obtained for the unidirectional and bidirectional models, shown in table 4.9. However, in contrast to multiple layers, increasing the complexity of the model is disadvantageous. The bidirectional models have a slight disadvantage over the unidirectional models.

		Unidirectional			**Bidirectional training**		
Buoy	Lead time (h)	RMSE (m)	sMAPE (%)	PCC	RMSE (m)	sMAPE (%)	PCC
414046	1	0.00	0.1	0.00	0.00	0.1	0.00
	3	-0.01	-0.5	-0.01	-0.01	-0.4	-0.01
	6	0.01	0.1	0.01	0.01	0.2	0.01
	12	0.04	0.8	0.03	0.04	0.8	0.03
	24	0.06	1.6	0.09	0.05	1.4	0.08
	48	0.06	0.7	0.14	0.05	0.5	0.14
414047	1	-0.01	0.0	-0.02	-0.01	-0.1	-0.00
	3	-0.02	-1.0	-0.00	-0.02	-1.2	-0.01
	6	-0.00	-1.1	0.00	-0.01	-1.3	0.00
	12	0.00	-2.0	0.02	-0.01	-2.3	0.01
	24	-0.01	-5.3	0.07	-0.03	-5.7	0.06
	48	-0.11	-12.5	0.15	-0.16	-13.8	0.14

Table 4.9: The average accuracy metrics for the deep recurrent models using uni- and bidirectional training. Note that positive values indicate improved performance.

Learning rates

In table 4.10 the results for the learning rates are given. The accuracy for buoy 414046 is consistent over the learning rates, where 10^{-5} has a slight improvement for a 48-hour forecast. For buoy 414047, the performance is considerably worse. The first two learning rates show a decrease in accuracy compared to the persistence model, and the last learning rate matches its accuracy.

		$\gamma = 10^{-3}$			$\gamma = 10^{-4}$			$\gamma = 10^{-5}$		
Buoy	Lead time (h)	RMSE (m)	sMAPE (%)	PCC	RMSE (m)	sMAPE (%)	PCC	RMSE (m)	sMAPE (%)	PCC
414046	1	0.01	0.3	0.00	0.01	0.3	0.00	-0.00	-0.1	-0.00
	3	-0.00	-0.2	-0.00	-0.00	-0.2	-0.00	-0.02	-0.9	-0.01
	6	0.02	0.2	0.02	0.02	0.3	0.02	0.01	-0.1	0.01
	12	0.04	0.8	0.03	0.03	0.7	0.04	0.04	0.8	0.03
	24	0.05	1.4	0.09	0.04	1.0	0.09	0.07	2.2	0.08
	48	0.04	-0.2	0.14	0.04	-0.5	0.14	0.09	2.6	0.14
414047	1	-0.01	-0.2	-0.01	0.00	0.3	-0.02	-0.01	-0.2	-0.00
	3	-0.02	-1.1	-0.02	-0.01	-0.6	0.00	-0.03	-1.7	-0.00
	6	-0.01	-1.1	-0.00	0.00	-0.8	0.01	-0.01	-1.8	0.00
	12	-0.01	-1.8	0.01	0.00	-2.0	0.03	0.00	-2.6	0.02
	24	-0.05	-4.9	0.05	-0.03	-6.6	0.07	0.01	-5.0	0.06
	48	-0.25	-12.4	0.13	-0.16	-18.2	0.15	0.02	-8.9	0.14

Table 4.10: The average accuracy metrics over all timestamps relative to the persistence model, for varying learning rates γ 10^{-3}, 10^{-4} and 10^{-5}. Note that positive values indicate improved performance.

Model architectures

The results for the ten different architectures, as discussed in section 4.3.1, are shown in table 4.12. For buoy 414046, the sMAPE slightly improves with model complexity. Nevertheless, overall the architectures seem to have similar accuracy. However, for buoy 414047, the opposite effect is visible for the sMAPE, model complexity seems to increase the error. A similar pattern can be seen for the PCC.

Best performing models

The results indicate that stepped input outperforms the successive input models. However, all results are averaged and give no insights about outliers. Therefore the distributions of average error over all time steps are considered. The insights are discussed here, and the distributions can be seen in the appendix in figs. C.1 and C.2. All the models using the successive input are outperformed by the persistence model for all three accuracy metrics. Therefore the models using successive inputs will be excluded from further analysis.

The LSTM model indicated better performance than the other recurrent models. If we consider the distributions (appendix figs. C.3 to C.5) we can still see RNN and GRU models perform better then the baseline. However, LSTM models show a greater number

Buoy	Model No.	RMSE (m)	sMAPE (%)	PCC	Model No.	RMSE (m)	sMAPE (%)	PCC
414046	1	-0.00	-0.7	0.03	6	0.05	1.4	0.05
	2	0.00	-0.5	0.04	7	0.02	0.0	0.05
	3	0.04	1.2	0.04	8	0.04	0.8	0.05
	4	0.02	-0.0	0.04	9	0.03	0.9	0.05
	5	0.02	-0.0	0.04	10	0.02	0.2	0.04
414047	1	-0.04	-4.3	0.03	6	0.00	-1.4	0.04
	2	-0.05	-5.1	0.04	7	0.01	-2.4	0.04
	3	-0.01	-1.5	0.04	8	-0.08	-7.2	0.04
	4	-0.01	-3.3	0.04	9	-0.08	-5.6	0.03
	5	-0.01	-3.3	0.04	10	-0.05	-5.5	-0.00

Table 4.11: The average accuracy metrics for the deep recurrent models using different model architectures, according to table 4.3. Note that positive values indicate improved performance.

of models improving the baseline, while the RNNs and GRUs have a long tail which includes models with poor performance.

Now let us consider the best performing models consistent with the majority of LSTM builds. Therefore we consider models with a smaller RMSE of 0.33 for both buoys, which is an average performance gain over the baseline of 0.04 and 0.07 for 414046 and 414047. In fig. 4.9, an overview of the frequency of all model variables is given. All shown architectures have an average RMSE in the range of $\{0.33, \ldots, 0.30\}$

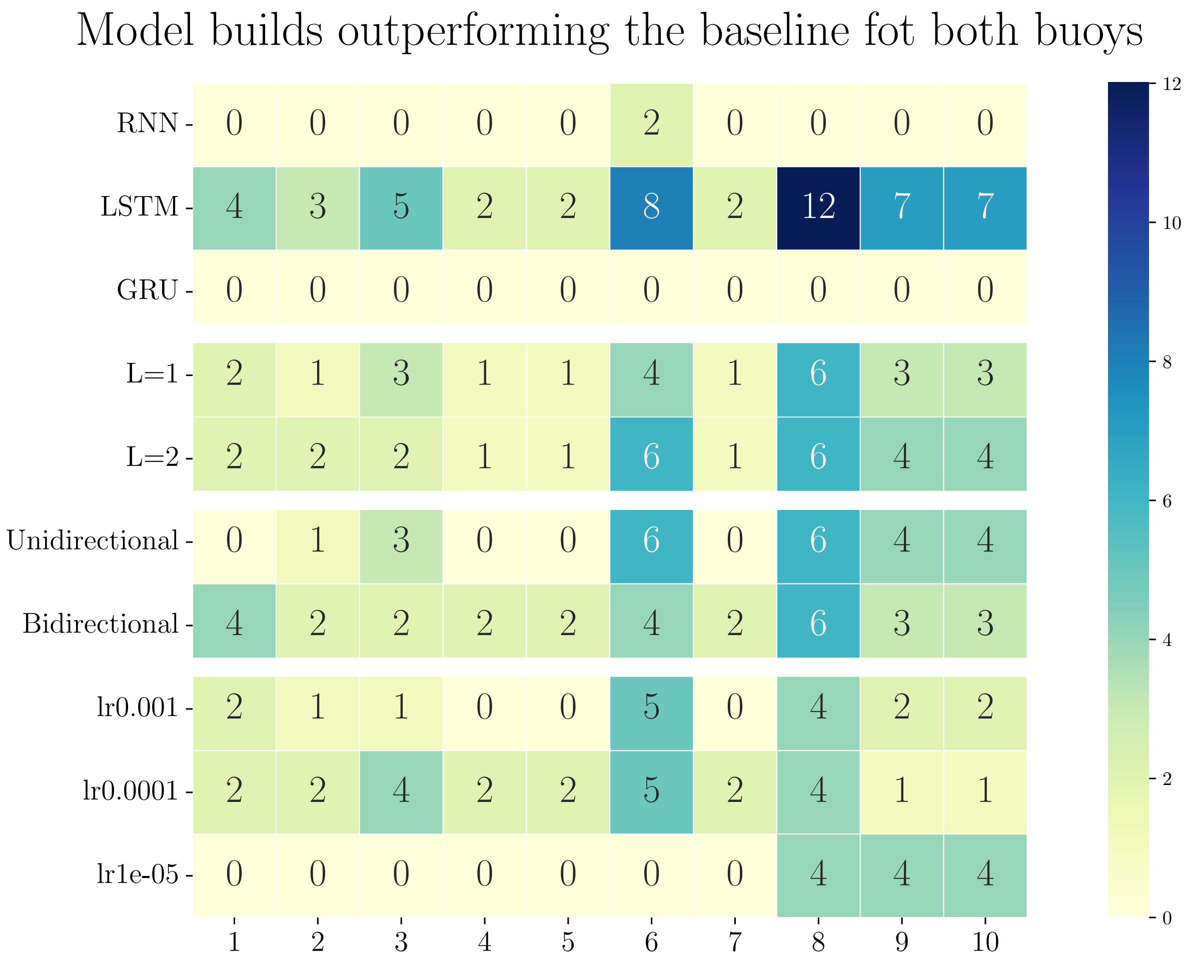
As we could expect from earlier results, the RNNs and GRUs are outperformed by the LSTMs. All variations on the LSTM model are included for architecture eight. Furthermore 6, 9 and 10 are also well represented within the top models. The amount of layers and directional training does not seem to impact the models' outcome. A lower learning rate seems only to increase performance for more complex models.

Let us consider the average results produced by these models, shown in table 4.12. The models seem to perform better on all accuracy metrics. There is a clear trend of improving accuracy with an increased forecasting horizon relative to the persistence model. However, it is salient that the three-hour forecast performs slightly worse.

	414046			414047		
Lead time	RMSE (m)	Smape (%)	PCC	RMSE (m)	Smape (%)	PCC
3	-0.00	-0.2	-0.00	0.00	-0.4	0.00
6	0.05	1.3	0.03	0.04	0.4	0.01
12	0.12	3.8	0.07	0.11	2.6	0.05
24	0.20	8.1	0.18	0.23	6.4	0.13
48	0.27	11.2	0.30	0.38	10.8	0.30

Table 4.12: The average accuracy metrics relative to the persistence model, for best performing models (RMSE < 0.33). Different models are used for each buoy No. and accuracy metric. Note that positive values indicate improved performance.

Now let us consider a 24 and 48-hour forecast by a model outperforming the baseline,



	1	2	3	4	5	6	7	8	9	10
RNN	0	0	0	0	0	2	0	0	0	0
LSTM	4	3	5	2	2	8	2	12	7	7
GRU	0	0	0	0	0	0	0	0	0	0
L=1	2	1	3	1	1	4	1	6	3	3
L=2	2	2	2	1	1	6	1	6	4	4
Unidirectional	0	1	3	0	0	6	0	6	4	4
Bidirectional	4	2	2	2	2	4	2	6	3	3
lr0.001	2	1	1	0	0	5	0	4	2	2
lr0.0001	2	2	4	2	2	5	2	4	1	1
lr1e-05	0	0	0	0	0	0	0	4	4	4

Figure 4.9: A quantitative heatmap for models with a RMSE < 0.33, outperforming the baseline for buoy 414046 and 414047. A total of 54 model architectures are shown. Model types and parameters are given on the y-axis and the architectures are given on the x-axis.

given in fig. 4.10. In the figure, it becomes clear that the models' forecast closely mimics the actual values, with a time lag taken into account. The behaviour is equal for the 24 and 48-hour plots, similar to the persistence model; however, some deviations cause a slight performance increase.

In fig. 4.11, we plot the same time interval with all forecasts from a single point in time. We can observe a slight deviation between the persistence and the LSTM model. The persistence model can only produce a straight line, while the LSTM first shows small deviations and then converges to a straight line, which deviates from the last seen point. No clear pattern can be seen when the model tends to increase or decrease the convergence value in contrast to the last seen data point.

4.4.2 Multivariate forecasting

For the multivariate models, experiments are run with a LSTM model using architecture eight with ($L \in \{1, 2\}$) and uni- and bidirectional training. The results are averaged and shown in table 4.13. All models, including individual performances, do not show any improvements over the baseline. Compared to the univariate architecture, the performances are even worse.

	414046			**414047**		
Input features	RMSE (m)	Smape (%)	PCC	RMSE (m)	Smape (%)	PCC
{SWH, GST}	0.0	-0.1	-0.0	0.01	-0.5	0.0
{SWH, GST, APD}	0.0	-0.4	-0.0	-0.03	-4.1	0.0
{SWH, GST, APD, WSPD}	0.0	-0.2	-0.0	-0.03	-4.2	-0.0

Table 4.13: The average accuracy metrics for a three-hour forecast relative to the persistence model, for multivariate LSTMs using architecture eight with ($L \in \{1, 2\}$) and uni- and bidirectional training. Note that positive values indicate improved performance.

4.5 Interpretation of the results

The search for model parameters gave some insights into what models produce the best accuracy. First, we found that the LSTM model got the best accuracy. Moreover, the effect of the input sequence was quite considerable. Furthermore, the model architectures did not significantly impact the accuracy, since all architectures could produce similar errors. However, it must be noted that this was averaged over all other model parameters. If we excluded another parameter, we could see that architecture eight in combination with an LSTM cell performed the best. In this architecture, all builds performed similarly. Despite some models producing better results, the accuracies were similar, and within a margin wherein the error difference was almost negligible.

With an increase up to 46 %, 37 % and 53 % for buoy 414046, at first sight the results seem quite significant. However, these numbers are treacherous due to their relativity and should be put into perspective of the baseline. The persistence model's prediction is not usable for forecasting in general, being unable to forecast any future change in slope. Secondly, an improvement can only be seen for larger lead times, indicating some averaging effect. The deep models are not as susceptible to gradient changes, while these

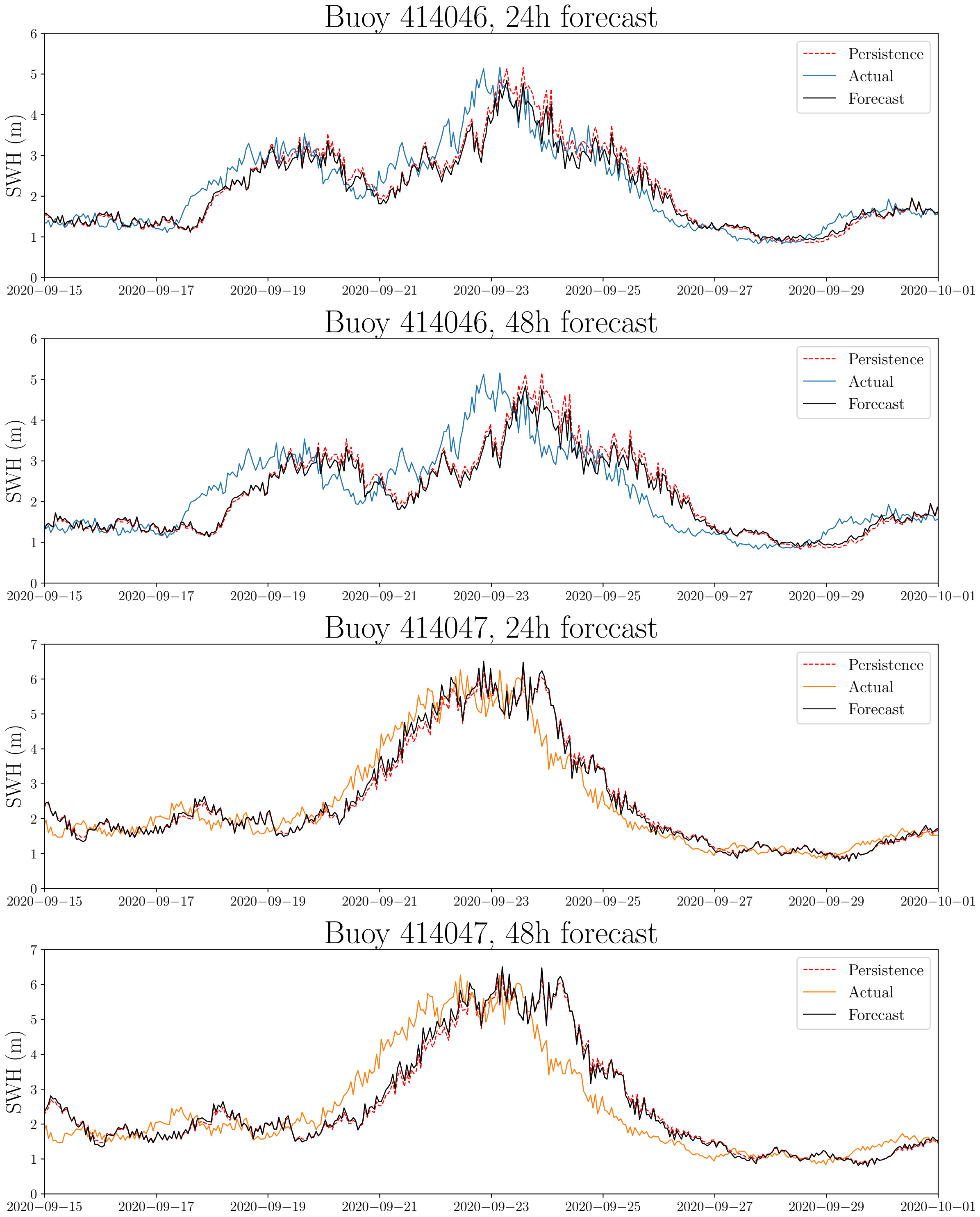


Figure 4.10: 24 and 48-hour forecast for the second semi-month of September 2020 for buoy 414046 and 414047. Using a singular unidirectional LSTM cell with architecture eight and a learning rate of 0.0001.

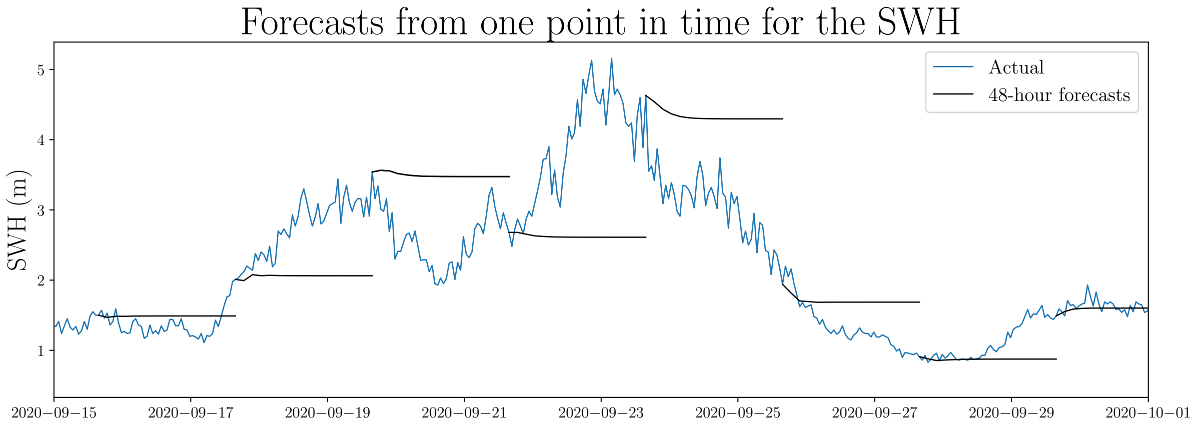


Figure 4.11: Multiple 48-hour forecasts from a single point in time for buoy 414046. Using a singular unidirectional LSTM cell with architecture eight and a learning rate of 0.0001.

dominate the persistence algorithms' accuracy. The persistence algorithm performs worse for a more erratic signal, which can be seen in the performance at buoy 414047. Since deep models are not as susceptible, the performance increase is more significant at buoy 414047.

Considering the differences seen in fig. 4.10, we could see the forecast and the baseline roughly following the same trend. Minor deviations made by the model cause increased performance. The model seems to recognise some patterns and slightly deviates the value of convergence as we saw in fig. 4.11. This deviation is small, and we could not conclude a structural convergence value. However, this effect is similar to averaging the signal and likely causes the LSTM model to produce the best performance. The model with the best long-term memory profits most from this strategy; therefore, the models also profit most from a stepped input.

The multivariate models show no improvements and even a decrease in accuracy compared to the univariate models. Furthermore, these models are more computationally demanding. They can only produce a single forecast. Therefore, these models seem less interesting in forecasting the SWH, and the focus will shift solely to univariate models in the following chapters.

Chapter 5

Transfer learning for the significant wave height forecast to enhance model performance

As seen in chapter 4, various deep recurrent models were not capable of forecasting the significant wave height (SWH). A possible reason for poor model performance is the lack of training data. Transfer learning is a technique that is often used to overcome this problem. Although the results in the previous chapter suggest that increasing the amount of training data does not lead to better model performance, this chapter aims to explore this issue further. Transfer learning will be used to improve model performance or exclude a lack of data as a possible cause for bad model performance. The experimental setup is discussed in section 5.1, the results in section 5.2 and finally these results are interpreted in section 5.3.

5.1 Experimental setup

For all experiments in this chapter, a singular unidirectional long short term memory (LSTM) cell with architecture eight (table 4.3), a stepped input and a learning rate of 0.0001 is used. First, the data usage of current models is explored by forecasting using different training sizes. The model is trained on sets of $\{1, \dots, 24\}$ months of data, the size for the validation and test sets are kept equal to the previous chapter, respectively 20 % and 10 % of the total size.

In figure fig. 5.1 the time average (3, 6, 12, 24 and 48 hour) outcome for all the root mean squared error (RMSE), the symmetric mean absolute percentage error (sMAPE) and the Pearson correlation (PCC) are given for all training sets. Using two months of data, the models can produce similar results as if trained on the whole training set. This indicates that the lack of data does not cause the bad performance of the models.

To confirm this indication, we run experiments with transferring information between the models. There are roughly two ways to apply transfer learning for a recurrent deep model; the weights of the recurrent cell and the weights of the fully connected layer could be transferred. These parts of the model consist of multiple weight matrices, as discussed in section 4.1, and the matrices could be transferred individually. However, we restrict our search to transferring the whole recurrent cell or the fully connected layer.

For these experiments, we train the model on the dataset from a source buoy. The recurrent cell or the fully connected layer for this trained model is then frozen, so they

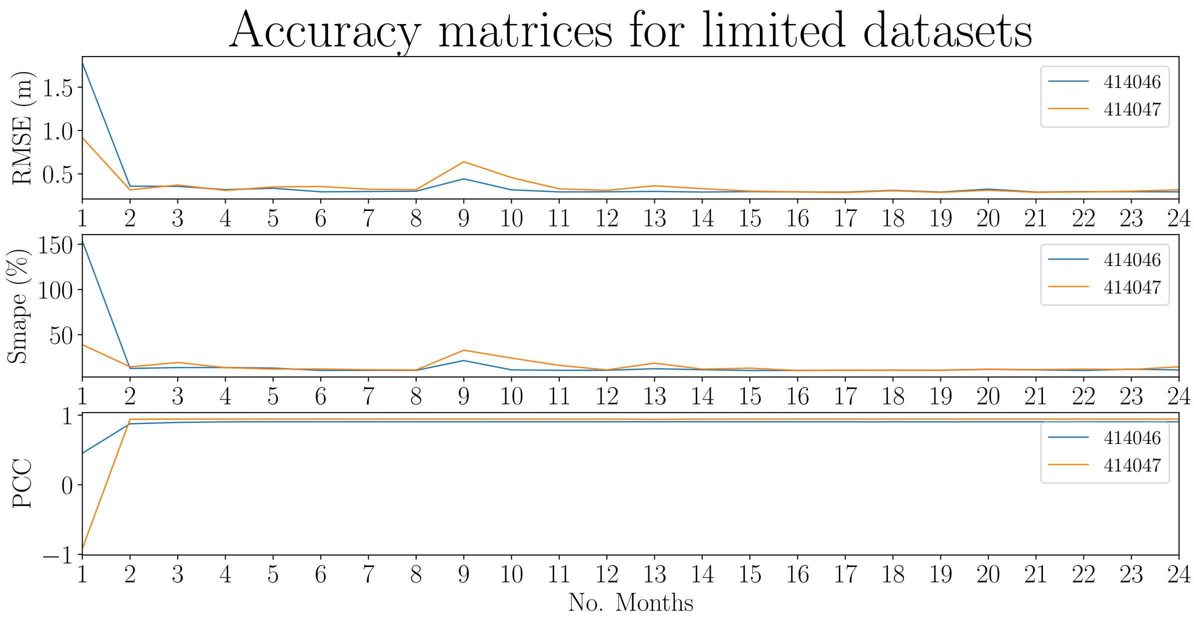


Figure 5.1: Average accuracy for a model trained with limited datasets. The no. months displays how many months of training data is used for the models. A singular unidirectional LSTM cell with architecture eight and a learning rate of 0.0001

do not gain gradient updates via backpropagation if trained. This model is then fine-tuned (trained) on the dataset of the target buoy with a patience of 5 epochs. We also experiment with freezing the whole model, which means no training for the target buoy. These experiments are run for 414046 as a source buoy and 414047 as a target buoy, and vice versa. The results are compared for the average RMSE, sMAPE and PCC over forecasting horizons 3, 6, 12, 24 and 48. For all experiments, further training details and data preparation is kept the same as section 4.3.

5.2 Results

The model trained and tested on the source buoy is used as a baseline for these experiments. These results are given in the appendix in table C.1. The results discussed are relative to this baseline in that positive values represent improvement concerning the persistence model. Note that negative zeros are given due to rounding.

The results for transferred models can be seen in table 5.1. The transferred models show no or negligible improvement over the original models. Furthermore, the differences between all transfer methods are negligible.

5.3 Interpretation of the results

All obtained models in chapter 4 closely mimic the last seen value with some small deviations. In this chapter, we have shown that the models could reproduce this behaviour with small amounts of training data. This implies that increasing the available data would not improve the current models. We reinforced this implication by experimenting with three different weight transfers over the models. All three proposed transfer methods did

		Fine-tuned layers								
		Neither			**Recurrent cell**			**Fully connected**		
Source buoy ↓ Target buoy	Lead time (h)	RMSE (m)	sMAPE (%)	PCC	RMSE (m)	sMAPE (%)	PCC	RMSE (m)	sMAPE (%)	PCC
414046 ↓ 414047	3	-0.00	0.4	-0.0	-0.00	0.3	-0.0	-0.00	0.4	-0.0
	6	0.00	1.3	-0.0	0.00	1.2	-0.0	0.00	1.5	-0.0
	12	0.00	2.1	-0.0	0.00	1.5	-0.0	0.00	2.3	-0.0
	24	-0.01	1.6	-0.0	-0.01	1.0	-0.0	0.01	1.9	-0.0
	48	-0.02	0.9	-0.0	-0.02	0.8	-0.0	0.01	1.0	-0.0
414047 ↓ 414046	3	0.00	0.0	-0.0	-0.00	0.0	-0.0	0.00	0.0	-0.0
	6	0.01	0.0	-0.0	0.01	0.0	-0.0	0.01	0.1	-0.0
	12	0.01	0.1	-0.0	0.01	0.1	-0.0	0.01	0.2	-0.0
	24	0.02	0.4	-0.0	0.02	0.3	-0.0	0.02	0.4	-0.0
	48	0.03	0.8	-0.0	0.02	0.6	-0.0	0.03	0.8	-0.0

Table 5.1: The accuracy metric for transferred models for buoy 414046 and 414047 in both directions. Three different scenarios for weight freezing are applied. The model is initially trained on the source buoy's data, and then the fine-tuned layers are trained on the target buoy's data. The accuracy metrics are relative to the baseline of table C.1.

not improve the accuracy, and the transferred models showed the same behaviour as the initial models.

The performance, the behaviour and the limited data necessary to learn this behaviour suggest the deep recurrent models cannot find and learn relevant correlations within the data. Therefore, the model finds a similar behaviour to the persistence model, which could be applied to all signals. Forecasting the SWH using deep recurrent models does not seem to be a problem of data quantity.

Chapter 6

Signal decomposition for the significant wave height to enhance model performance

In previous chapters, we found that various deep recurrent models could not produce a viable forecast for the significant wave height (SWH) and that the amount of data is not the bottleneck. The models do not seem to find any underlying correlations in the historical data. We could help the model interpret the data through data manipulations to enhance performance. The simplest known class of examples are preprocessing methods such as normalisation. In this chapter, empirical mode decomposition (EMD) is analysed and used to assist the interpretability of the data by decomposing the signal into multiple signals with lower frequency. First EMD and its complications are discussed in section 6.1, then the experimental setup is discussed in section 6.2. Finally, we give the results in section 6.3 and interpret them in section 6.4.

6.1 Empirical mode decomposition

EMD is essentially defined by an algorithm and dynamically derives basis functions from a signal. The algorithm recursively finds these basis functions from high to low frequency, known as sifting. The functions resulting from the 'sifting' algorithm are called intrinsic mode functions (IMFs). The outcome of the sifting process depends on local data characteristics and is not pre-determined as in wavelet transformations [48].

The sifting algorithm of EMD can be seen as six individual steps that lead to the sifting of signal $x(t)$, such that

$$x(t) = \sum_{i=1}^{n} \text{IMF}_i(t) + r(t), \tag{6.1}$$

where $r(t)$ is the residual and IMF_i is i^{th} intrinsic mode. The steps taken are given below:

1. Identify all the extrema of signal $x(t)$.
2. Interpolate between minima and the maxima, resulting in two envelopes $e_{\min}(t)$ and $e_{\max}(t)$.
3. Compute the mean envelope $m(t) = \frac{e_{\min}(t)+e_{\max}(t)}{2}$.

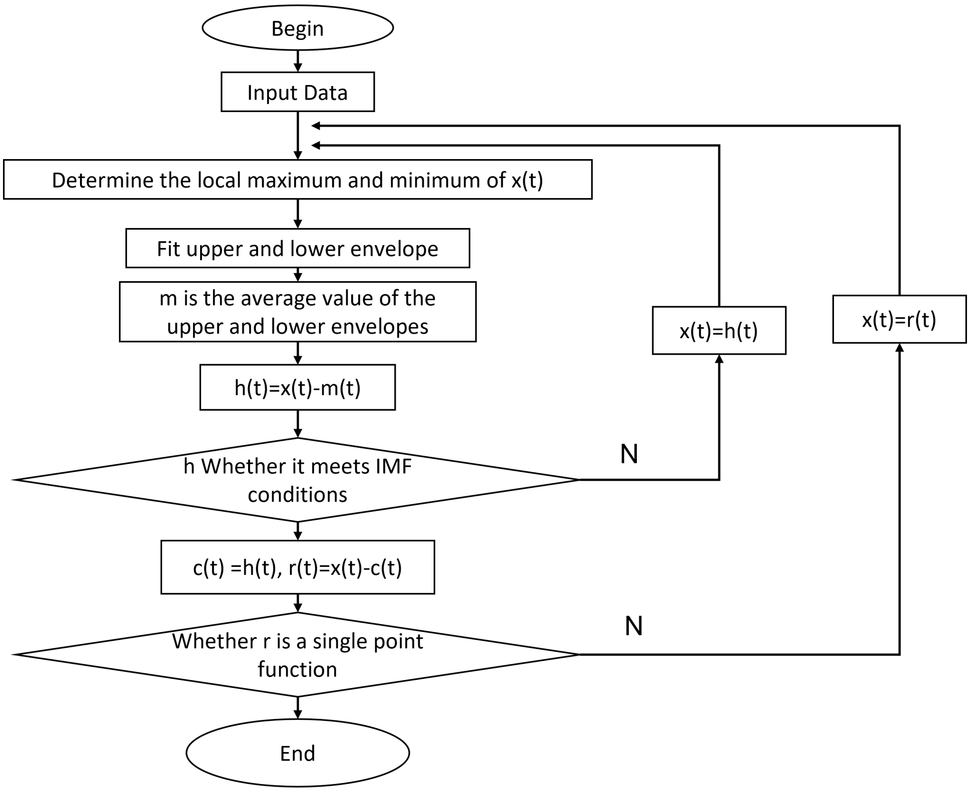

Begin
Input Data
Determine the local maximum and minimum of x(t)
Fit upper and lower envelope
m is the average value of the upper and lower envelopes
h(t)=x(t)-m(t)
h Whether it meets IMF conditions
N
x(t)=h(t)
c(t) =h(t), r(t)=x(t)-c(t)
Whether r is a single point function
N
x(t)=r(t)
End

Figure 6.1: A graphical representation of the sifting process for the EMD algorithm. $x(t)$ is the signal starting the sifting procedure, $m(t)$ is the mean envelope, $h(t)$ is the candidate IMF and $c(t)$ is a valid IMF function.

4. Subtract this mean envelope from the original signal, to obtain a candidate IMF $h(t) = x(t) - m(t)$.

5. If $h(t)$ does not satisfy the criteria of an IMF, repeat above steps on $h(t)$.

6. If $h(t)$ satisfies the criteria, the first IMF is found, depicted as $c(t) = h(t)$. And we repeat the process on the residual $r(t) = x(t) - c(t)$.

7. This iteration is stopped when the stopping criteria are reached, and we have a residual $r(t)$ left.

A flowchart of the algorithm described above can be seen in fig. 6.1.
Two criteria determine whether the candidate IMF $h(t)$ is a valid IMF.

1. The number of zero-crossings and extrema may differ by at most one.

2. $m(t)$ must be close to or exactly zero, according to some set criterion ϵ_c.

The sifting process will stop when the residual $r(t)$ is a monotonic or constant function where we cannot subtract an IMF. Or when the standard deviation is between 0.2 and 0.3

Any interpolation method would suffice; however, cubic splines are preferred since they minimise the amount of sifts [52, 68]. Other methods tend to "overdecompose" the signal and spread characteristics over multiple IMFs.

Empirical mode decomposition applied to forecasting

Since EMD is a dynamic process and there is no analytical formulation available, we can not make a theoretical analysis of the performance (beforehand). Evaluation requires simulation experiments, and the outcome will depend on the data itself [68]. Experimenting for forecasting tends to be even more difficult since the number and behaviour for each IMF varies per decomposed signal.

When applying EMD, caution is necessary to prevent data leakage from the test set to the train set. EMD needs to be applied after splitting the dataset; otherwise, EMD does not produce valid results. EMD subtracts an envelope based on all the maxima and the minima in a signal. The distance between these extrema varies with a high probability for every sifting iteration. Therefore the interpolated line can extend into the test set. Furthermore, the cubic spline interpolation is applied over multiple points, which could lead to overlapping splines between the train and test set. However, this leakage is not always consistently prevented in literature [24, 25, 26, 27, 28, 29, 30]. This has led to the unsubstantiated conclusions that EMD could be used for significant improvement to the model's performance.

The effect of this data leakage for buoy 414046 can be seen in fig. 6.2. The figure shows three weeks of data divided into a test and train set. In this smaller set, the same effects can be seen as in the whole signal, and the difference for the earlier IMFs are easier to observe. In the appendix, in fig. D.1, a similar effect can be seen for the whole dataset.

For earlier IMFs, the differences between the sets are smaller and are mainly caused by the interpolation boundaries. For later IMFs, the differences become more significant, e.g. IMF three, where a gap can be seen between the train and test set. Especially the residual effect is of influence; these values have the largest weight, almost equal to all summed IMFs. The residual for the whole set shows a gradual increase of the SWH, while the training residual indicates a decrease for the last week.

In the example, the decomposed signals do not contain the same amount of IMFs. Let us consider the number of siftings necessary for subsets of 414046 and 414047. The subsets are created with a step size of three, $\{3k|k \in [0, 367]\}$, with the length approximately equal to the statistical significance of lagged versions (1.5 months) as found in section 4.2. The results over the training set are given in table 6.1.

	No. IMFs			
Buoy	**5**	**6**	**7**	**8**
414046	725	13 609	6893	18
414047	677	14 520	6017	31

Table 6.1: The number of IMFs obtained for the training data, using a sequence length of 1100 and a step size of three.

The variance for the number of IMFs is small for the SWH signals, and we can see that six sifting iterations are the most frequent. This variability in the number of IMFs causes extra complexity for deep forecasting models since these models generally presume a constant number of features. We can limit the number of siftings, which affects the outcome. Most likely, the residual's complexity will increase, increasing the difficulty for the models to interpret.

Since forecasting time series is a supervised learning problem, labelled targets are necessary. However, these target values cannot be included in the decomposed signal,

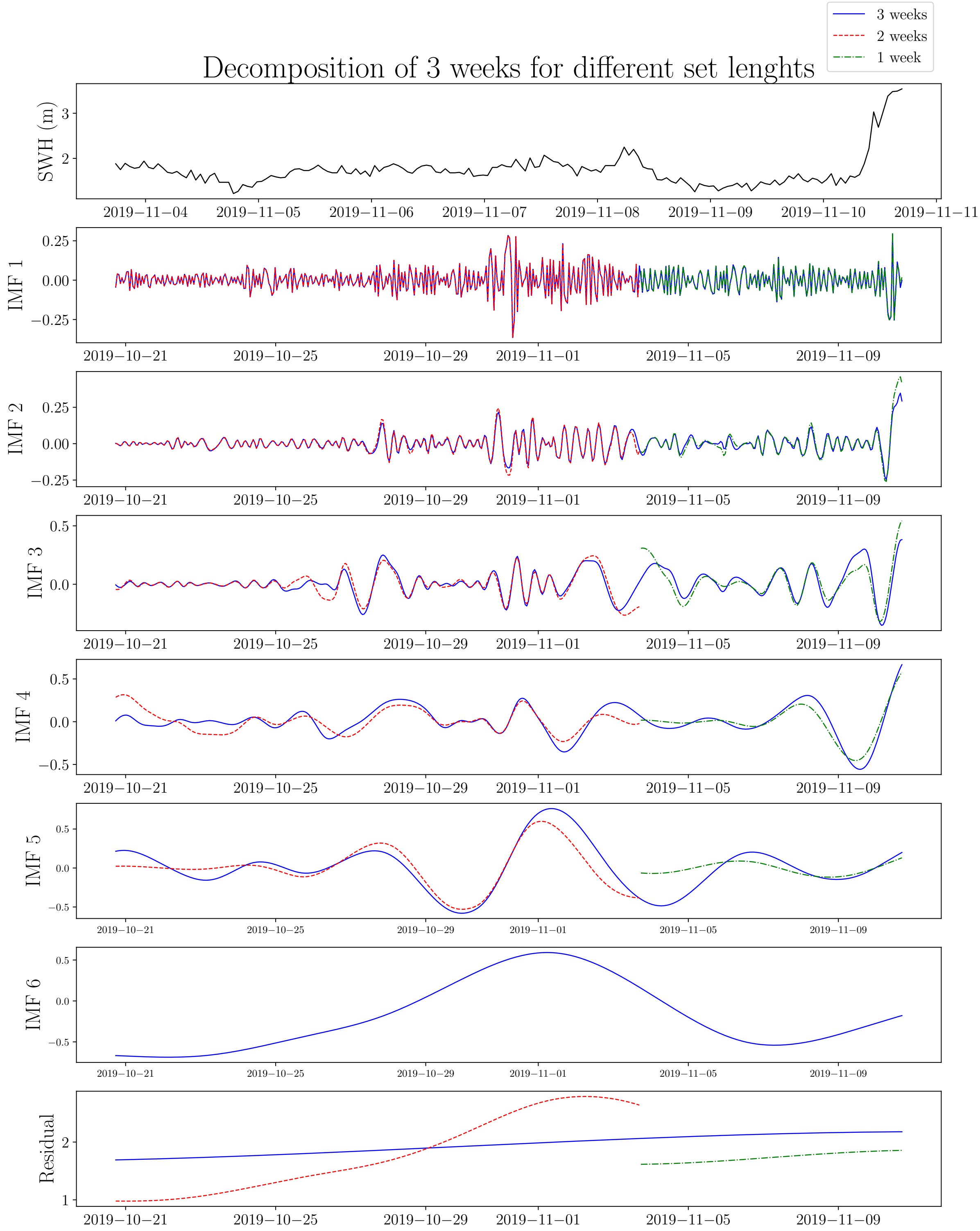


Figure 6.2: EMD for the three weeks, a training set (two weeks) and a test set (one week) for buoy 414046 for the hourly SWH data. The black line represents the original signal, the blue line the decomposition of the three weeks, the red line the decomposition of the training set and the green line the decomposition of the test set.

since this will change the IMFs with high probability. The uncertainty is caused by the dynamic nature of EMD. Therefore, the successive value for each individual IMF is unknown. This causes that creating a labelled set is not straightforward. In the next section, two frameworks are proposed to create a supervised learning problem such that EMD is viable in a deep learning framework.

6.2 Experimental setup

We ran experiments using two frameworks to solve the supervised problem. The first framework uses the value of the original signal as the target value for each IMF. Using the SWH as the target value requires a bias addition in the models since the IMF values are lower than the SWH. For each individual IMF level a separate long short term memory (LSTM) is trained. Despite the bias, we intend the model to learn characteristics influencing the outcome. This format assumes sufficient subsets to construct training sets for each IMF. Table 6.1 showed a lack of training data for IMF eight, therefore we limited the EMD algorithm to seven siftings.

The second framework assumes that IMFs negligibly change if an extra data point is included. This framework is introduced to avoid the extra bias in framework one. For this procedure we applied EMD two times; over the sets $\{1, \ldots, t\}$, $\{1, \ldots, t+1\}$. The last value $t+1$ is then extracted as a target value for each IMF. To forecast each individual IMF an LSTM will be trained separately. This method could lead to a different amount of IMFs per decomposed set. If this occurs, EMD will be limited to the lowest number IMFs. Besides this limit, the sifting algorithm is overall limited to seven.

Both frameworks produce a forecast for each IMF, where we should average the IMFs for the first framework and sum for the second for a final forecast. However, in both procedures, variations are made such that the IMFs may be skewed. Therefore all outcomes are fed into a neural network (NN) to correct the error to produce a SWH forecast. In fig. 6.3, a graphical representation of both frameworks is given. The figure also depicts the training and target sets for the deep models.

The outcome produced by proposed frameworks relies on several assumptions. These are quite restrictive and could have a significant impact on the outcome. These assumptions are:

1. Each IMF contains predictive characteristics for future values of the SWH.
2. IMFs of the same order, IMF^s i and IMF^k i, of signal s and signal k, contain similar predictive characteristics.
3. higher IMFs can be summed to a single IMF, where similar predictive characteristics can be found.

Framework two relies on another assumption that the decomposed signals for the signal $\{1, \ldots, t\}$ and $\{1, \ldots, t+1\}$ are similar.

Note that multivariate recurrent models would be able to process all the IMFs together to produce an outcome for every IMF. However, these models include the relations between all input variables (IMFs). Therefore, univariate models are chosen in the discussed framework to reduce the complexity of the recurrent cells.

For these experiments singular unidirectional LSTM cells are used, and we use architecture eight (table 4.3). The NNs, which map the IMFs into a single value, consists of one fully connected layer of seven neurons.

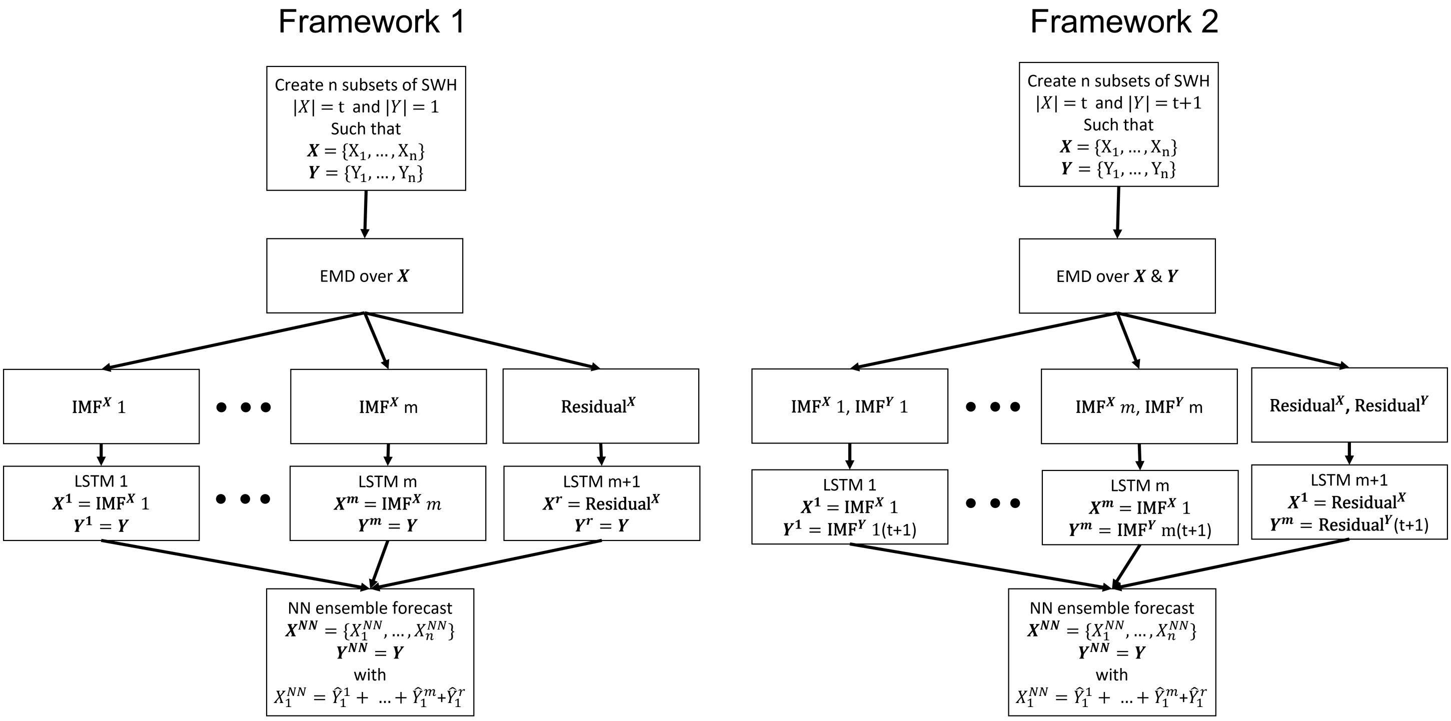

Framework 1
Create n subsets of SWH
$|X| = $ t and $|Y| = 1$
Such that
$\boldsymbol{X} = \{X_1, \dots, X_n\}$
$\boldsymbol{Y} = \{Y_1, \dots, Y_n\}$
EMD over $\boldsymbol{X}$
IMF^X 1
IMF^X m
Residual^X
LSTM 1
$\boldsymbol{X}^1 = \text{IMF}^X\ 1$
$\boldsymbol{Y}^1 = \boldsymbol{Y}$
LSTM m
$\boldsymbol{X}^m = \text{IMF}^X\ m$
$\boldsymbol{Y}^m = \boldsymbol{Y}$
LSTM m+1
$\boldsymbol{X}^r = \text{Residual}^X$
$\boldsymbol{Y}^r = \boldsymbol{Y}$
NN ensemble forecast
$\boldsymbol{X}^{NN} = \{X_1^{NN}, \dots, X_n^{NN}\}$
$\boldsymbol{Y}^{NN} = \boldsymbol{Y}$
with
$X_1^{NN} = \hat{Y}_1^1 + \dots + \hat{Y}_1^m + \hat{Y}_1^r$
Framework 2
Create n subsets of SWH
$|X| = $ t and $|Y| = $ t+1
Such that
$\boldsymbol{X} = \{X_1, \dots, X_n\}$
$\boldsymbol{Y} = \{Y_1, \dots, Y_n\}$
EMD over $\boldsymbol{X}$ & $\boldsymbol{Y}$
IMF^X 1, IMF^Y 1
IMF^X m, IMF^Y m
Residual^X, Residual^Y
LSTM 1
$\boldsymbol{X}^1 = \text{IMF}^X\ 1$
$\boldsymbol{Y}^1 = \text{IMF}^Y\ 1(t+1)$
LSTM m
$\boldsymbol{X}^m = \text{IMF}^X\ 1$
$\boldsymbol{Y}^m = \text{IMF}^Y\ m(t+1)$
LSTM m+1
$\boldsymbol{X}^1 = \text{Residual}^X$
$\boldsymbol{Y}^m = \text{Residual}^Y(t+1)$
NN ensemble forecast
$\boldsymbol{X}^{NN} = \{X_1^{NN}, \dots, X_n^{NN}\}$
$\boldsymbol{Y}^{NN} = \boldsymbol{Y}$
with
$X_1^{NN} = \hat{Y}_1^1 + \dots + \hat{Y}_1^m + \hat{Y}_1^r$

Figure 6.3: Graphical representation two EMD deep learning frameworks. X is the training set, Y is the target set, n depicts the number of samples, and m is the number of IMFs. The superscript depicts for which model the sets are created and over which set EMD is applied. The subscript depicts the set number and timestamp.

Data preparation

For the EMD, we use data with a time-lag up to 1100 hours. Which matches the statistical significance found in section 4.2 for both buoys. We use similar sizes for the train, validation and test set; 70 % 20 % and 10 %. However, the validation and test set are lengthened with 1100, to ensure we forecast the same time interval as in previous chapters' results.

With these splits we create samples with $\{3k | k \in [0, 367]\}$. The step size is already introduced before the EMD algorithm is applied to increase the number of training samples. Furthermore, we exclude sequences that have missing values in the last 48 hours. To make a training dataset, the samples will be decomposed by EMD. For the second framework, proposed EMD is applied two times similarly to framework one over the sets $\{1, \dots, t\}$, $\{1, \dots, t+1\}$. The last value for each IMF$(t+1)$ from the decomposed set $\{1, \dots, t+1\}$ is set as the target value.

Training details & evaluation metrics

The training details and evaluation metrics are kept the same as in section 4.3. However, only one learning rate is used and set to 1×10^{-4}. Furthermore, we use the standard stopping criteria ϵ_c of 0.005 [69]. EMD is implemented using the package of Quinn et al. [69].

6.3 Results

As a naive baseline, the persistence algorithm is used, found in table 4.5. All results shown are relative to the persistence algorithm and can be seen in table 6.2. All evaluation metrics for framework one show decreased performances, except for a negligible improvement at 48 hours for buoy 414046. Framework two shows a significant improvement over framework one. However, this framework is still outperformed by the persistence model. Solely a small improvement can be seen at 48 hours.

		Framework one			Framework two		
Buoy	Lead time (h)	RMSE (m)	sMAPE (%)	PCC	RMSE (m)	sMAPE (%)	PCC)
414046	3	-0.56	-23.4	0.60	-0.25	-14.4	0.15
	6	-0.28	-14.8	-0.24	-0.27	-16.2	-0.24
	12	-0.20	-12.2	-0.25	-0.16	-10.4	-0.15
	24	-0.10	-5.8	-0.19	-0.04	-5.8	-0.09
	48	0.02	-0.9	-0.06	0.11	1.6	0.09
414047	3	-0.75	-29.1	0.67	-0.38	-23.3	0.18
	6	-0.44	-21.6	-0.27	-0.40	-23.3	-0.25
	12	-0.37	-19.2	-0.26	-0.27	-17.0	-0.17
	24	-0.24	-12.9	-0.27	-0.12	-12.3	-0.12
	48	-0.02	-6.4	-0.15	0.14	-3.2	0.10

Table 6.2: The accuracy metrics relative to the persistence model for the two EMD frameworks. Note that positive values indicate improved performance.

In fig. 6.4, framework two's 24 and 48-hour forecasts are given for the semi-month of September 2020, including both buoys. The behaviour of the forecast is locally erratic and underestimates the period of higher waves. Furthermore, the model seems to follow the slope direction last seen in the data.

In fig. 6.5, we plot the same time interval with all forecasts from a single point in time. We can see that the produced outcome looks erratic and varies greatly from the persistence model. The persistence model can only produce a straight line, while the EMD framework shows various slope changes. Initially, the model seems to follow the last slope of previous values. With a higher lead time, no clear pattern can be seen in the forecasted values. Similar behaviour can be observed for framework one in the appendix in fig. D.2.

6.4 Interpretation of the results

Both EMD frameworks are outperformed by the persistence algorithm. The models' predictions overshoot for the smallest lead time and then show erratic oscillatory behaviour. Furthermore, framework two follows the data trend with a greater time lag than the persistence model. Indicating the model is not as susceptible to changes in trend. Overall, both models could not produce valid forecasts for the SWH. Nevertheless, it is interesting to note that the proposed frameworks do not converge to a constant value and show multiple slope changes.

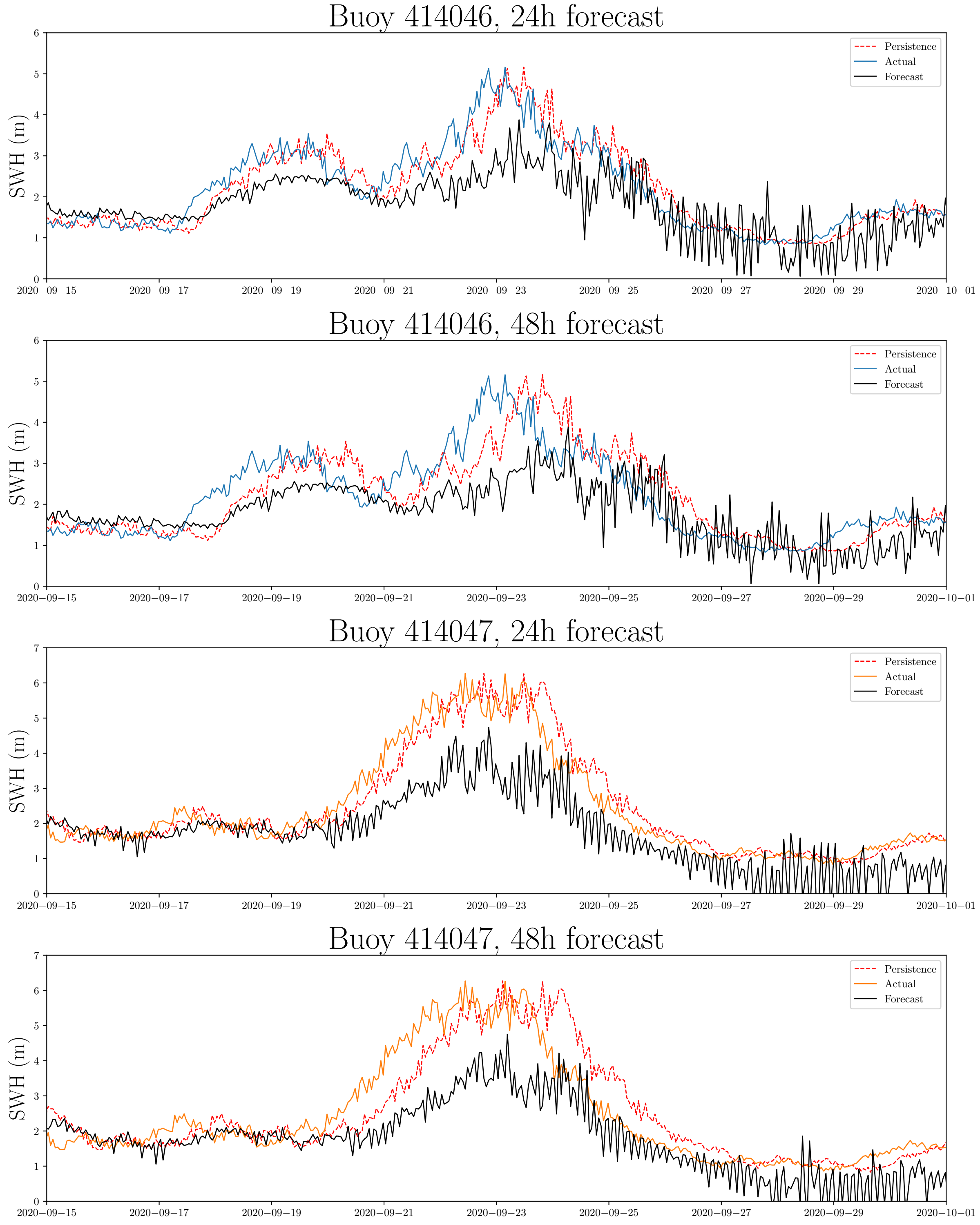


Figure 6.4: 24 and 48-hour forecast of the EMD framework two for the second semi-month of September 2020 for buoy 414046 and 414047.

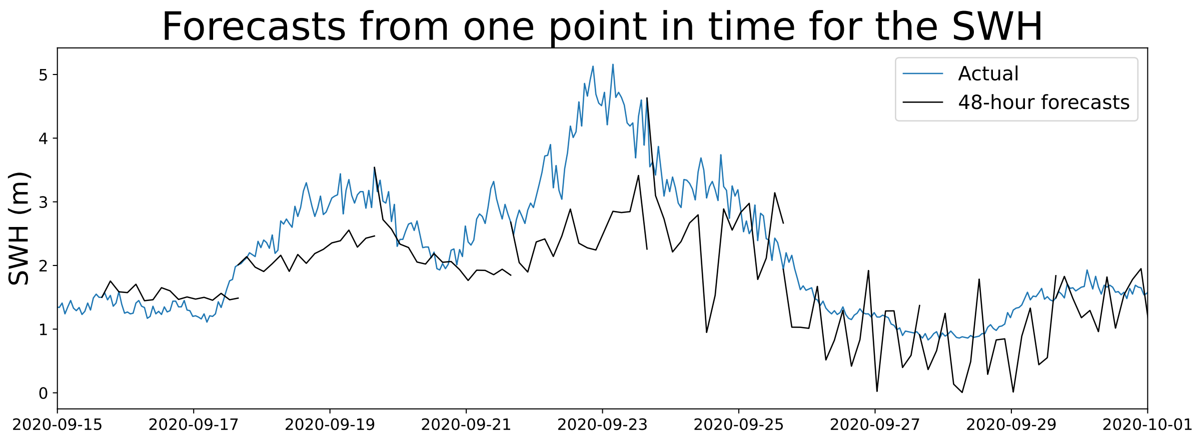


Figure 6.5: Multiple 48-hour forecasts from a single point in time for buoy 414046 using EMD framework two.

The difference between the two frameworks could be caused by "persistence" behaviour by the deep models. For framework one, it is not viable to forecast the last seen value since the LSTM has to overcome an extra bias. However, the second framework can adopt such behaviour. The variances for the summed IMFs of one-time step are bounded by the variance of similar time steps of the SWH. The max variance for one time step for the SWH is equal to the max variance for the difference between the last values of decomposed signals $\{1, \ldots, t\}$ and $\{1, \ldots, t+1\}$. This enables the second framework to adopt similar behaviour as the persistence model.

It must be noted that both frameworks are based on numerous assumptions. The results indicate that some of these assumptions lack validity and negatively impact the models' performance. Besides the fact that we do not know the predictive capabilities of EMD, we expect that the predictive characteristics are not similar for the same level IMFs. When the sea state is calm EMD will meet the stopping criteria earlier than rough behaviour. Therefore, the frequency of similar levels of IMF will vary due to the state of the sea. In section 4.2, we observed non-stationary behaviour for monthly subsets. This behaviour likely results in a greater variance in the distance between maxima and minima, which impacts the frequency of the produced IMFs. Which would cause the IMFs not to contain similar predictive characteristics and therefore "polluting" the training set. This pollution makes it even harder for the models to find relations in the data and causes decreased accuracy.

Chapter 7

Conclusion and discussion

This thesis investigated the possibilities of using deep learning frameworks to forecast the significant wave height (SWH). First we investigated the use of recurrent neural networks (RNNs), long short term memorys (LSTMs) and gated recurrent units (GRUs). We experimented with various model builds and considered two different methods to enhance forecasting. We experimented with transfer learning by leveraging information from other buoys and frameworks using empirical mode decomposition (EMD) to enhance the model's signal interpretation. This chapter concludes this thesis by summarising the work done and answering the research questions.

In chapter 4, we ran 720 experiments using different model builds on two datasets and compared them to the persistence algorithm. These experiments did not indicate that deep recurrent models alone would suffice to forecast the SWH using historical data. For some models, we saw increased accuracies for longer lead times for all accuracy metrics used; the root mean squared error (RMSE), the symmetric mean absolute percentage error (sMAPE) and the Pearson correlation (PCC).

Although these models produced "good" accuracy scores, they are not capable of forecasting the SWH. Slight deviations to the persistence behaviour achieved the increase in performance. Moreover, this increase should be put in perspective of the baseline; since this accuracy is the bare minimum, the models should outperform. The persistence algorithm cannot initiate any shift in trend; therefore, the error increases with every change. The deep recurrent models could not forecast any trend shift in our experiments, which limits their predictive capabilities.

The accuracy of these models highly depends on the modelling choices made and which dataset is used. Therefore it is impossible to draw definitive conclusions about the effectiveness of the models. However, the experiments were extensive, and all common variations were tested. Although we did not test all the same datasets as the papers [18, 19, 20], we are confident that the papers published on this subject overestimate the effectiveness of their models and wrongly suggest feasibility.

In chapter 5, transfer learning is considered to enhance model performance. We first analysed the behaviour of the current models for different training data sizes. These experiments showed that the models had already reached their maximum performance using two months of data. Applying three different weight transfers did not increase both datasets' performance and displayed the same behaviour. This suggested that the models cannot interpret the data correctly or that the historical relationship is insufficient to forecast the SWH. We found no indication that transfer learning could enhance the performance of current deep recurrent models by leveraging data from other locations.

Therefore, we could not analyse any geographical and meteorological features to select suitable source buoys.

In literature, various papers proposed an EMD framework to enhance the accuracy of the deep recurrent models. In these papers, EMD was applied before splitting the data. In chapter 6, we have shown that this method most likely led to information being leaked from the testing set to the training set, thus resulting in higher accuracy. We proposed two new EMD frameworks to prevent leakage, using two different target sets to train the models. Both frameworks failed to increase the performance of the LSTM models and even decreased the accuracy significantly. We found no indications substantiating the assumption that intrinsic mode functions (IMFs) contains predictive characteristics for future values of the SWH. Therefore we could not use EMD in a machine learning framework to enhance the forecasting of the SWH.

This work investigated different methods to forecast the SWH using deep learning frameworks. Although no successful method was found, we comprehensively analysed the current state of deep learning frameworks, which gave good insights into the standing of current literature for SWH forecasts. There are still many improvements and fine-tuning possible to extend this analysis. Some points of discussion and suggestions for further research are proposed below.

While we concluded that transfer learning is unsuitable for the current state-of-the-art deep recurrent models, this does not exclude the effectiveness of future frameworks. While literature currently does not provide any effective deep models for forecasting the SWH, future research may enable this method to enhance accuracy. Finally, we only experimented with three weight transfers. One could experiment with more sophisticated transfers by transferring specific model components.

In this thesis, solely the final forecast was evaluated and discussed for the deep EMD frameworks. A more extensive evaluation of the models' behaviour could give more insight into the predictive capabilities of EMD. For example, analysing the forecasts produced by the LSTMs for each IMF. Furthermore we did not experiment with any variations to EMD, e.g. ensemble empirical mode decomposition (EEMD) and complete ensemble empirical mode decomposition (CEEMD), which could be more promising [70].

Both EMD frameworks used are based on multiple assumptions necessary to enable the production of a supervised learning set. For future research, we suggest treating this as an unsupervised problem to drop the assumptions necessary to produce a supervised training set. This can be achieved by training a new (multivariate) model for each decomposition and producing unsupervised forecasts. This means that training, validation, and test sets can be retrieved after applying EMD for each decomposition. A newly trained model's accuracy and confidence interval can then be obtained via bootstrapping.

In this thesis, we comprehensively investigated deep learning frameworks to forecast the SWH. Although many papers have been published about this subject, we could not find a deep learning framework capable of producing a usable SWH forecast. Nevertheless, we gave future studies a good starting point and research direction.

Appendix A

NDBC measurement data

Description	Unit
Station number	-
Coordinates	°
Site elevation (above sea level)	m
Air temperature height	°C
Anemometer height (above site elevation)	m
Barometer elevation (above sea level)	m
Sea temperature depth (below water line)	m
Water depth	m
Watch circle radius	yd

Table A.1: Buoy Data by the National Data Buoy Centre [1].

Quantity	Abbreviation	unit	Description
Wind direction	WDIR	°	Wind direction (the direction the wind is coming from in degrees clockwise from true north) during the same period used for WSPD.
Wind speed	WSPD	$\mathrm{m\,s^{-1}}$	Wind speed averaged over eight minutes for buoys and two minutes for land stations. Reported hourly.
Gust speed	GST	$\mathrm{m\,s^{-1}}$	Peak five or eight-second gust speed (measured during the eight-minute or two-minute period).
Significant wave height	WVHT	m	Significant wave height is calculated as the average of the highest one-third of all wave heights during a 20-minute sampling period.
Dominant wave period	DPD	s	Dominant wave period is the period with the maximum wave energy.
Average wave period	APD	s	Average wave period of all waves during a 20-minute sampling period.
Mean wind direction	MWD	°	The direction from which the waves at the dominant period (DPD) are coming. The units are degrees from true north, increasing clockwise, with north as 0 (zero) degrees and East as 90 degrees.
Sea level pressure	PRES	hPa	Sea level pressure for C-MAN sites and Great Lakes buoys.
Air Temperature	ATMP	°C	Air temperature for sensor heights on buoys.
Sea surface temperature	WTMP	°C	Temperature of the water at the sea surface.
Dewpoint temperature	DEWP	°C	Dewpoint temperature is taken at the same height as the air temperature measurement.
Visibility	VIS	M (1852 m)	Station visibility. Note that buoy stations are limited to reports from 0 to 1.6 M.
Pressure tendency	PTDY	hPa	Pressure Tendency is the direction (plus or minus) and the amount of pressure change for a three-hour period ending at the time of observation.
Tide	TIDE	ft	The water level in feet above or below Mean Lower Low Water.

Table A.2: Standard Meteorological Data and their abbreviations by the National Data Buoy Centre [1].

Quantity	Abbreviation	unit	Description
Significant wave height	WVHT	m	Significant Wave Height is the average height (meters) of the highest one-third of the waves during a 20-minute sampling period.
Swell height	SwH	m	Swell height is the vertical distance between any swell crest and the succeeding swell wave trough.
Swell Period	SwP	s	Swell Period is the time that it takes successive swell wave crests or troughs to pass a fixed point.
Wind Wave Height	WWH	m	Wind Wave Height is the vertical distance (meters) between any wind-wave crest and the succeeding wind-wave trough (independent of swell waves).
Wind Wave Period	WWP	s	Wind Wave Period is the time that it takes successive wind-wave crests or troughs to pass a fixed point.
Swell wave direction	SwD	°	The direction from which the swell waves at the swell wave period (SWPD) are coming. The units are degrees from true north, increasing clockwise, with north as 0° and East as 90°.
Wind waves direction	WWD	°	The direction from which the wind waves at the wind wave period (WWPD) are coming. The units are degrees from true north, increasing clockwise, with north as 0° and East as 90 degrees.
Steepness	STEEPNESS	-	Wave steepness is the ratio of wave height to wavelength and indicates wave stability. When wave steepness exceeds a 1/7 ratio; the wave becomes unstable and begins to break.
Average wave period	APD	s	Average Wave Period is the average period (s) of the highest one-third of the wave observed during a 20 min sampling period.
Mean wave direction	MWD	°	The direction from which the waves at the dominant period (DPD) are coming. The units are degrees from true north, increasing clockwise, with north as 0 (zero) degrees and East as 90°. See the Wave Measurements section.

Table A.3: Detailed Wave Data and their abbreviations by the National Data Buoy Centre [1].

Quantity	Abbreviation	unit	Description
Depth	-	m	Depth at which measurements are taken.
Ocean Temperature	OTMP	°C	The direct measurement of the Ocean Temperature (as opposed to the indirect measurement (see WTMP above)).
Conductivity	COND	$\mu S\,cm^{-1}$	Conductivity is a measure of the electrical conductivity properties of seawater.
Salinity	SAL	-	Salinity is computed by a known functional relationship between the measured electrical conductivity of seawater (CON), temperature (OTMP) and pressure. Salinity is computed using the Practical Salinity Scale of 1978 (PSS78) and reported in Practical Salinity Units.
Oxygen Concentration	O2	%	Dissolved oxygen as a percentage.
Oxygen Concentration	O2PPM	-	Dissolved oxygen in parts per million.
Chlorophyll Concentration	CLCON	$\mu g\,L^{-1}$	Chlorophyll concentration.
Turbidity	TURB	-	Turbidity is an expression of the optical property that causes light to be scattered and absorbed rather than transmitted in straight lines through the sample (APHA 1980). Units are Formazine Turbidity Units (FTU).
pH	PH	-	A measure of the acidity or alkalinity of the seawater.
Eh	EH	mV	Redox (oxidation and reduction) potential of seawater

Table A.4: Oceanic data and their abbreviations by the National Data Buoy Centre [1].

Appendix B

Data analysis

Buoy: 414 046 **Depth**: 5549 m **location**: 23.822 °N 68.384 °W

Variables	**Minimum**	**Mean**	**Maximum**
Wind direction (WDIR) (°)	0	68.6	98.0
gust speed (GST) (m s^{-1})	0	5.9	28.8
significant wave height (SWH)(m)	0	1.7	9.4
gust speed (GST) (m s^{-1})	0	7.5	35.5
Dominant wave period (DPD) (s)	3.1	9.0	19.1
average wave period (APD) (s)	3.3	6.1	13.4
Mean wave direction (MWD) (°)	0	57.8	98.0
Sea level pressure (PRES) (hPa)	974.8	1018.0	1029.0
Air Temperature (ATMP (°C)	18.9	26.6	31.7
Sea surface temperature (WTMP) (°C)	18.7	27.4	30.9
Dewpoint temperature (DEWP) (°C)	11.1	22.4	28.4

Table B.1: Information and data characteristics from buoy 414046 for 2017 until 2020.

Buoy: 414 047 **Depth**: 5547 m **location**: 27.465 °N 71.452 °W

Variables	**Minimum**	**Mean**	**Maximum**
Wind direction (WDIR) (°)	1.0	61.9	98.0
gust speed (GST) (m s^{-1})	0	5.8	26.7
significant wave height (SWH) (m)	0.47	1.7	12.3
gust speed (GST) (m s^{-1})	0	7.4	37.6
Dominant wave period (DPD, s)	3.3	8.9	19.1
average wave period (APD) (s)	3.8	6.2	11.7
Mean wave direction (MWD) (°)	0	54.7	98.0
Sea level pressure (PRES) (hPa)	986.1	1020.8	1035.3
Air Temperature (ATMP) (°C)	16.4	25.2	31.0
Sea surface temperature (WTMP) (°C)	21.8	26.2	32.4
Dewpoint temperature (DEWP) (°C)	6.1	20.3	27.1

Table B.2: Information and data characteristics from buoy 414047 for 2017 until 2020.

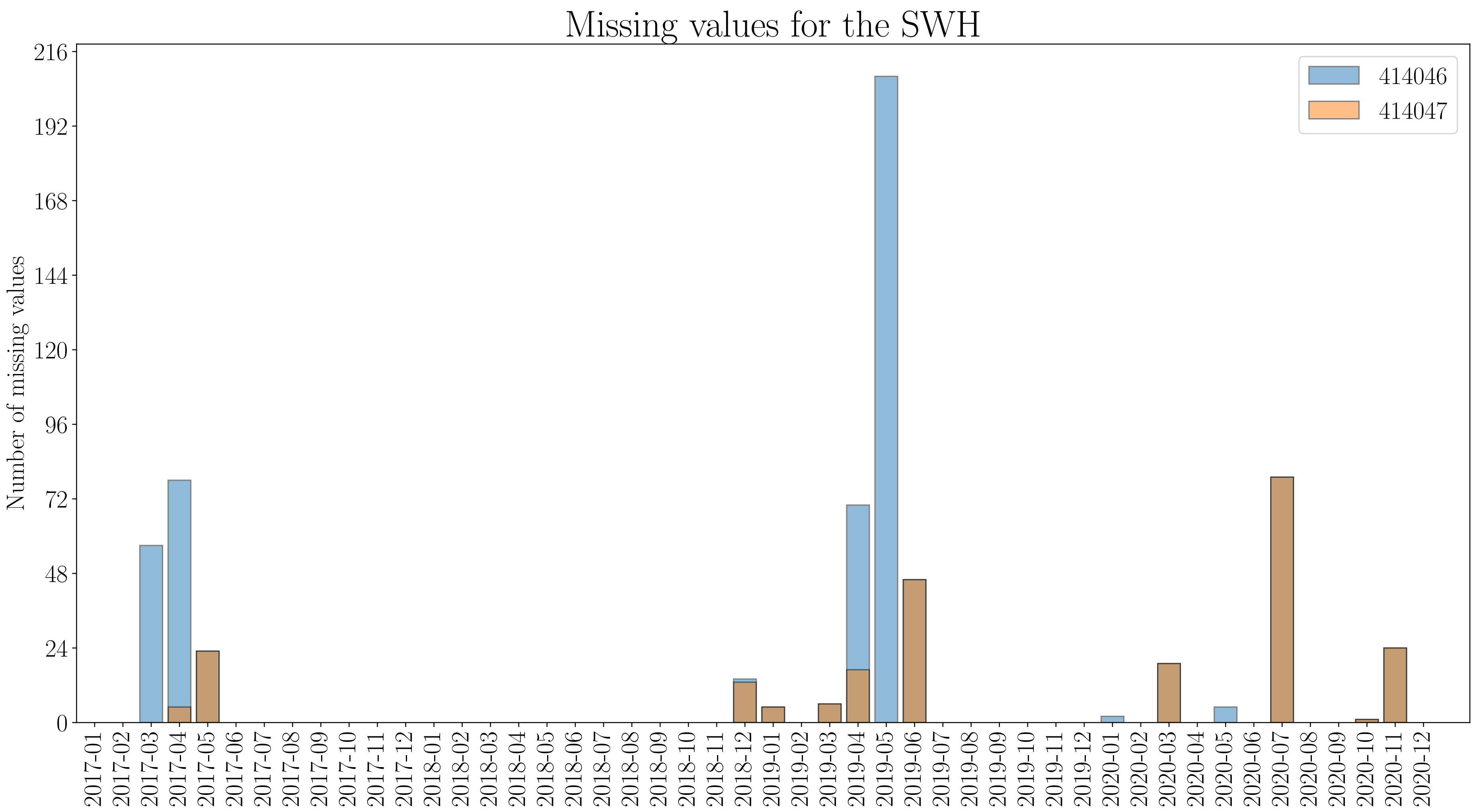


Figure B.1: The distribution of missing values for buoy 414046 and 414047 over the years 2017 until 2020. The histogram uses a bin size of one month.

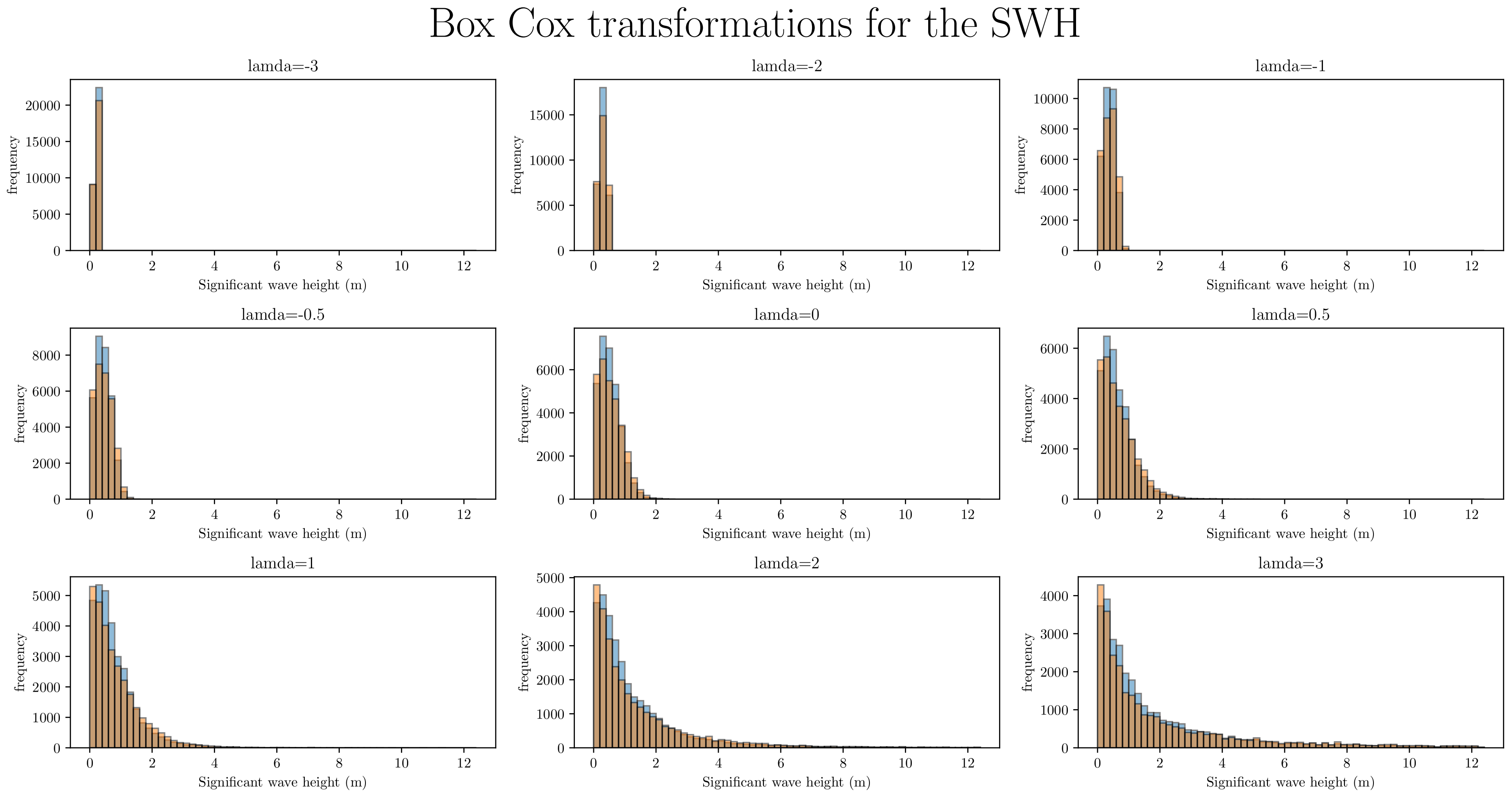


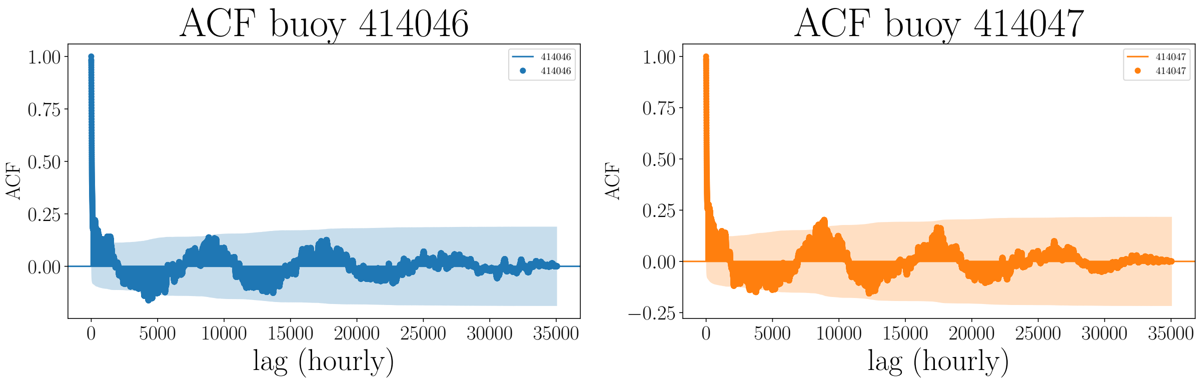
Figure B.2: Box-Cox transformations over Buoy 414046 and 414047 with lambda: −3, −2, −1, −0.5, 0, 0.5, 1, 2 and 3. The data is shifted with 0.01 to avoid values of 0. The histograms use a bin size of 0.1.

Lambda	Buoy	test statistic	P-value
-3	414 046	0.80	5.2×10^{-18}
	414 047	0.80	1.4×10^{-18}
-2	414 046	0.84	2.0×10^{-8}
	414 047	0.81	6.1×10^{-10}
-1	414 046	0.87	3.2×10^{-5}
	414 047	0.86	6.5×10^{-7}
-0.5	414 046	0.90	5.5×10^{-5}
	414 047	0.88	6.5×10^{-7}
0	414 046	0.92	4.8×10^{-5}
	414 047	0.90	1.0×10^{-6}
0.5	414 046	0.92	5.8×10^{-5}
	414 047	0.91	1.2×10^{-6}
1	414 046	0.92	5.0×10^{-5}
	414 047	0.90	1.0×10^{-6}
2	414 046	0.90	4.3×10^{-5}
	414 047	0.90	8.8×10^{-7}
3	414 046	0.87	3.8×10^{-5}
	414 047	0.87	7.8×10^{-7}

Table B.3: Normality tests for the significant wave height and Box-Cox transformations for Lambdas:-3, -2, -1, -0.5, 0, 0.5, 1, 2 and 3. The tests are conducted 100 times using 1000 random samples. Note that a lambda equal to one is the original distribution

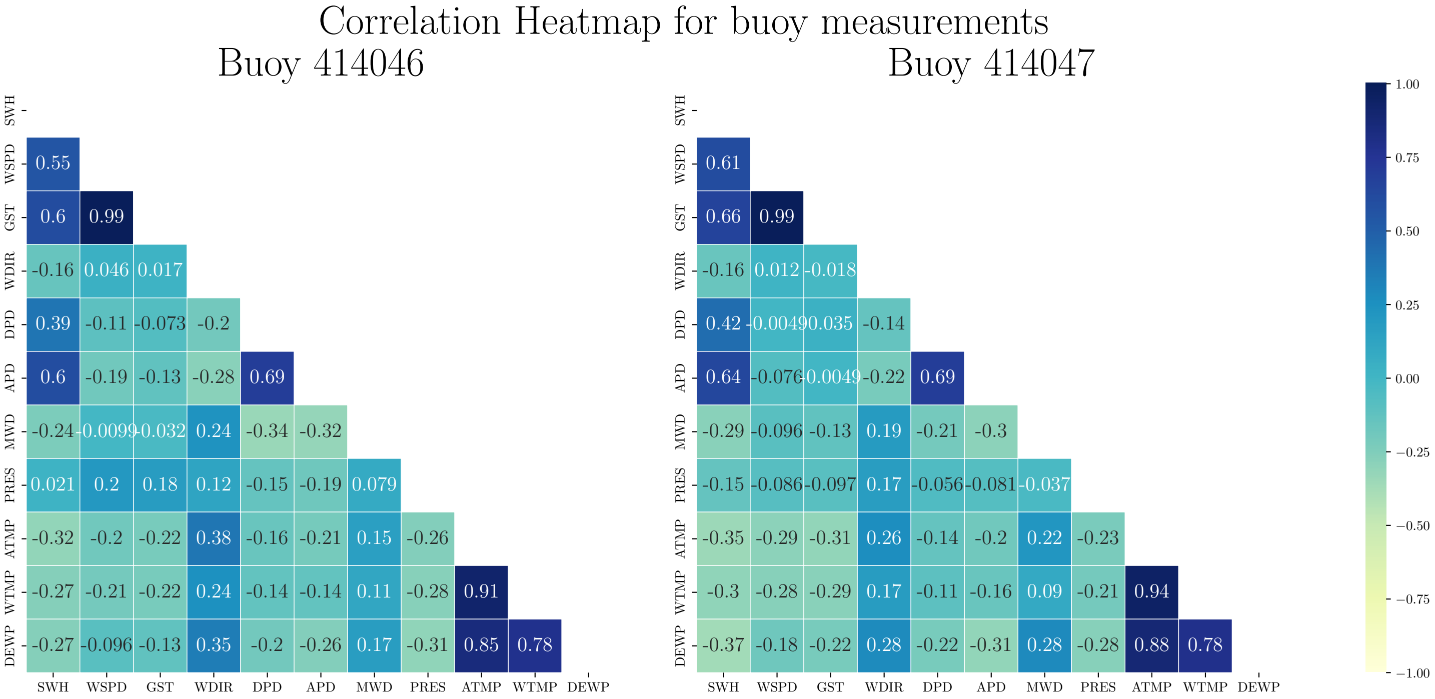
Period	Buoy	test statistic	P-value	NO of lags considered
4 years	414 046	-13.5	3.6×10^{-25}	49
	414 047	-14.2	1.7×10^{-26}	52
1 year	414 046	-8.2	7.0×10^{-10}	±24
	414 047	-7.6	9.3×10^{-10}	±34
6 months	414 046	-5.9	1.2×10^{-5}	±18
	414 047	-6.0	4.8×10^{-5}	±18
1 month	414 046	-2.8	0.11	±9
	414 047	-3.0	0.10	±8
1 week	414 046	-1.6	0.48	±4
	414 047	-1.7	0.44	±5

Table B.4: Augmented Dickey-Fuller tests for the significant wave height for buoys 414046 and 414047. The results given are averaged over a split of the given period over the full dataset. The number of lags is rounded into integers.


ACF buoy 414046

ACF buoy 414047

Figure B.3: The autocorrelation function (ACF) plot for the SWH for the whole signal of buoy 414046 and 414047. The coloured cone depicts the 95 % interval according to Bartlett's test and is an indicator of the significance threshold. Anything outside the cone can be seen as statistical significant.


Correlation Heatmap for buoy measurements

Buoy 414046

	SWH	WSPD	GST	WDIR	DPD	APD	MWD	PRES	ATMP	WTMP	DEWP
SWH											
WSPD	0.55										
GST	0.6	0.99									
WDIR	-0.16	0.046	0.017								
DPD	0.39	-0.11	-0.073	-0.2							
APD	0.6	-0.19	-0.13	-0.28	0.69						
MWD	-0.24	-0.0099	-0.032	0.24	-0.34	-0.32					
PRES	-0.021	0.2	0.18	0.12	-0.15	-0.19	0.079				
ATMP	-0.32	-0.2	-0.22	0.38	-0.16	-0.21	0.15	-0.26			
WTMP	-0.27	-0.21	-0.22	0.24	-0.14	-0.14	0.11	-0.28	0.91		
DEWP	-0.27	-0.096	-0.13	0.35	-0.2	-0.26	0.17	-0.31	0.85	0.78	

Buoy 414047

	SWH	WSPD	GST	WDIR	DPD	APD	MWD	PRES	ATMP	WTMP	DEWP
SWH											
WSPD	0.61										
GST	0.66	0.99									
WDIR	-0.16	0.012	-0.018								
DPD	0.42	-0.0049	0.035	-0.14							
APD	0.64	-0.076	-0.0049	-0.22	0.69						
MWD	-0.29	-0.096	-0.13	0.19	-0.21	-0.3					
PRES	-0.15	-0.086	-0.097	0.17	-0.056	-0.081	-0.037				
ATMP	-0.35	-0.29	-0.31	0.26	-0.14	-0.2	0.22	-0.23			
WTMP	-0.3	-0.28	-0.29	0.17	-0.11	-0.16	0.09	-0.21	0.94		
DEWP	-0.37	-0.18	-0.22	0.28	-0.22	-0.31	0.28	-0.28	0.88	0.78	

Figure B.4: A Heatmap for the correlation with the significant wave height for available measurement variables for buoy 414046 and 414047.

Appendix C

Model performances

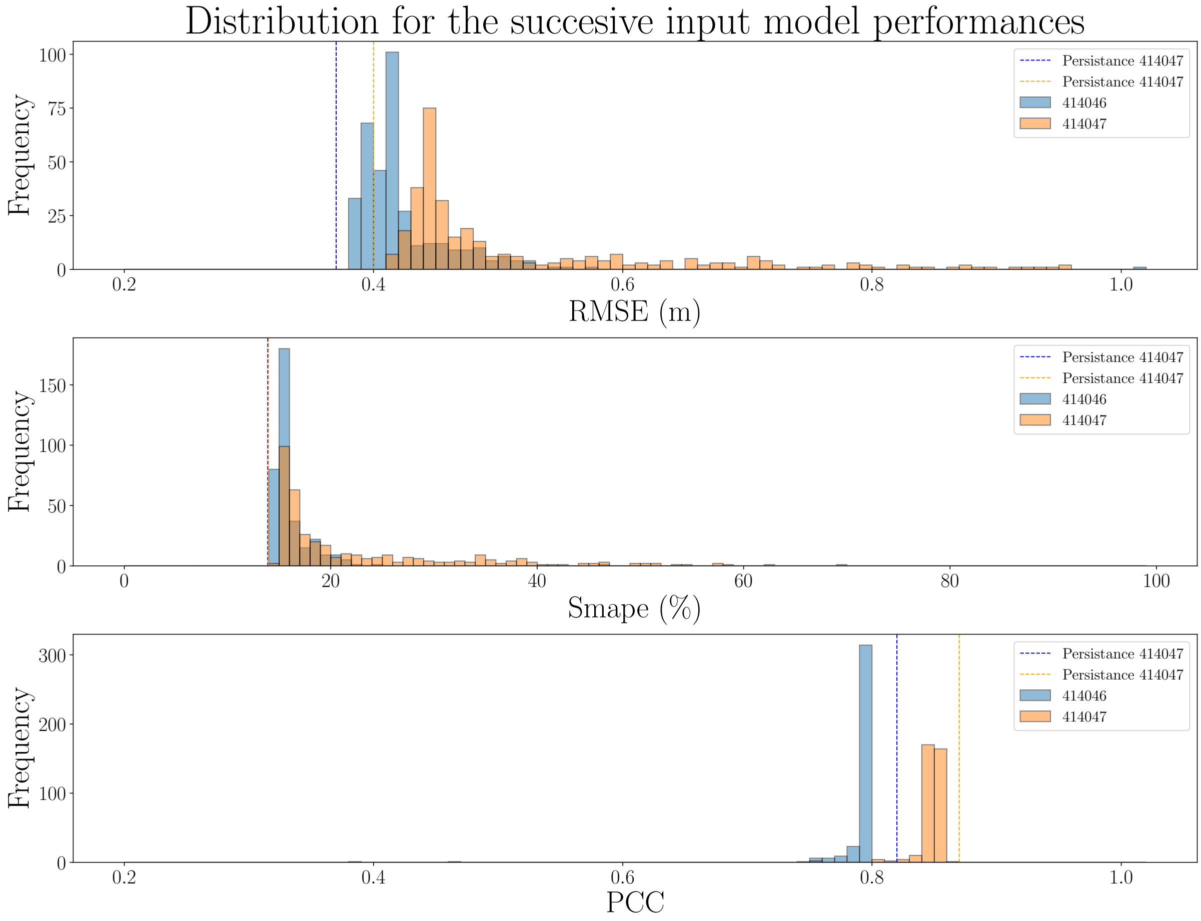


Figure C.1: The distribution of averaged performance over all time steps for all models using sequential input for buoy 414046 and 414047. The histograms use bin sizes of 0.01, 1 and 0.01 and the dashed line represents the accuracy of the persistence model. Note that smaller RMSE and sMAPE are desired, and a larger PCC.

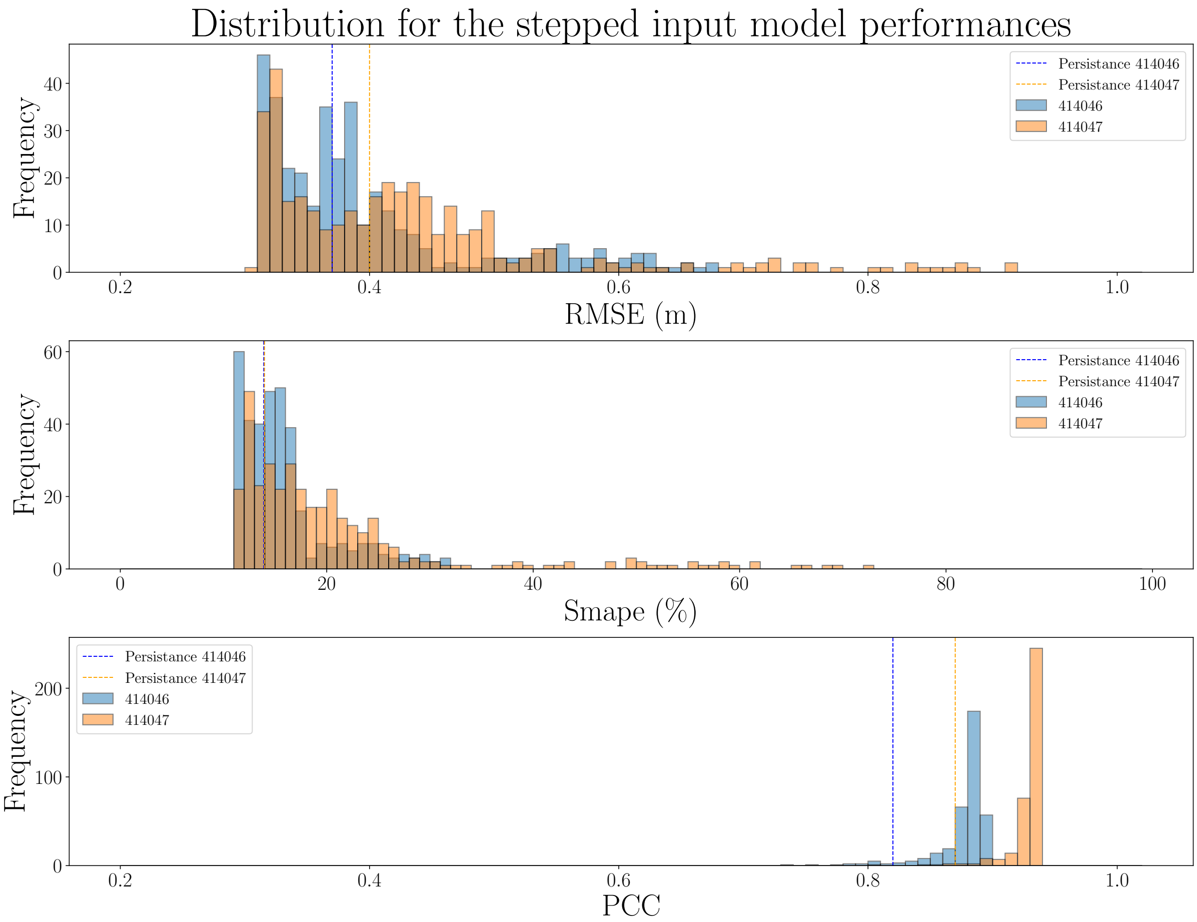


Figure C.2: The distribution of averaged performance over all time steps, for all models using stepped input for buoy 414046 and 414047. The histograms use bin sizes of 0.01, 1 and 0.01 and the dashed line represents the accuracy of the persistence model. Note that smaller RMSE and sMAPE are desired, and a larger PCC.

	414046			**414047**		
Lead time (h)	RMSE (m)	Smape (%)	PCC	RMSE (m)	Smape (%)	PCC
3	0.18	6.7	0.97	0.18	6.6	0.98
6	0.21	7.7	0.96	0.21	7.6	0.97
12	0.26	9.3	0.93	0.26	9.3	0.96
24	0.33	12.1	0.89	0.34	12.4	0.93
48	0.46	16.8	0.78	0.48	17.4	0.87

Table C.1: Accuracy for a singular unidirectional LSTM cell using architecture eight with a stepped input and a learning rate of 0.0001.

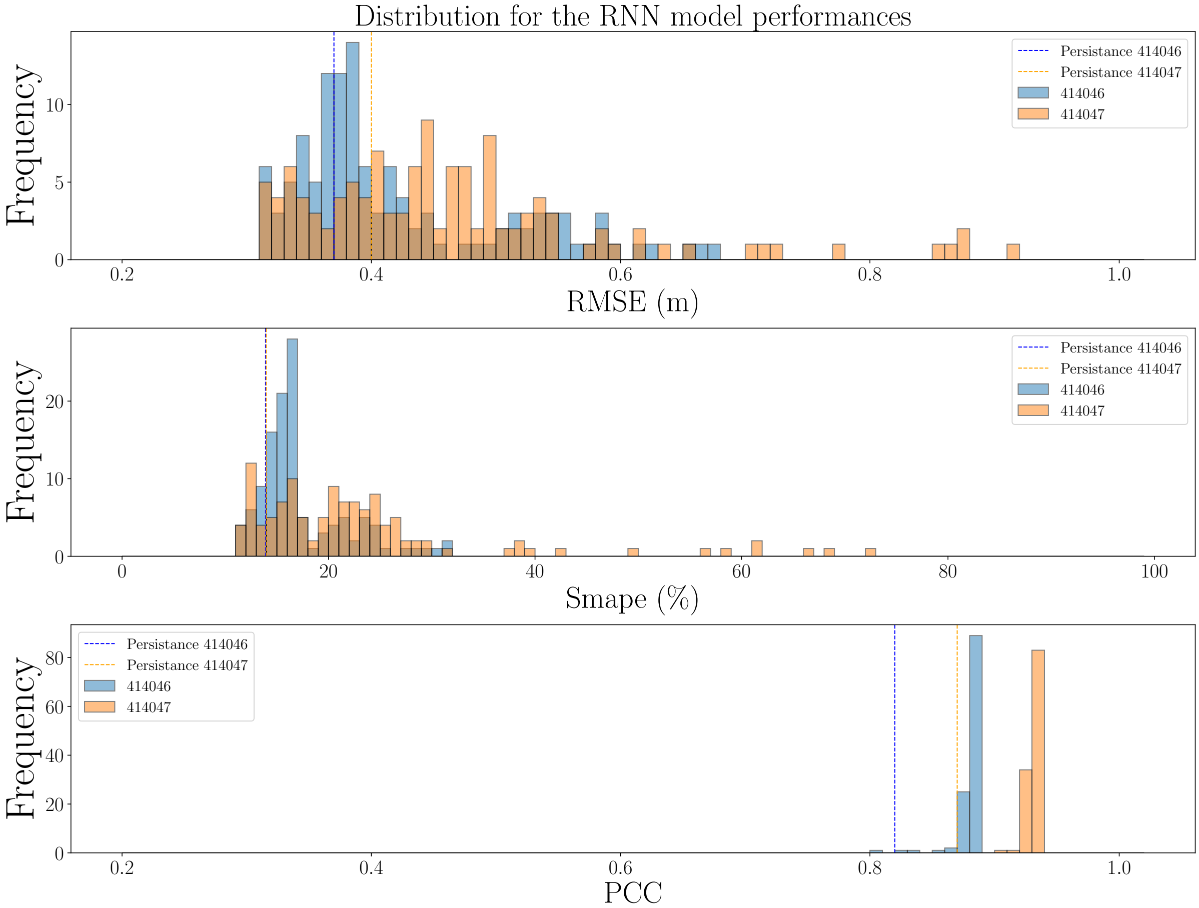


Figure C.3: The distribution of averaged performance over all time steps for the RNN models using stepped input for buoy 414046 and 414047. The histograms use bin sizes of 0.01, 1 and 0.01 and the dashed line represents the accuracy of the persistence model. Note that smaller RMSE and sMAPE are desired, and a larger PCC.

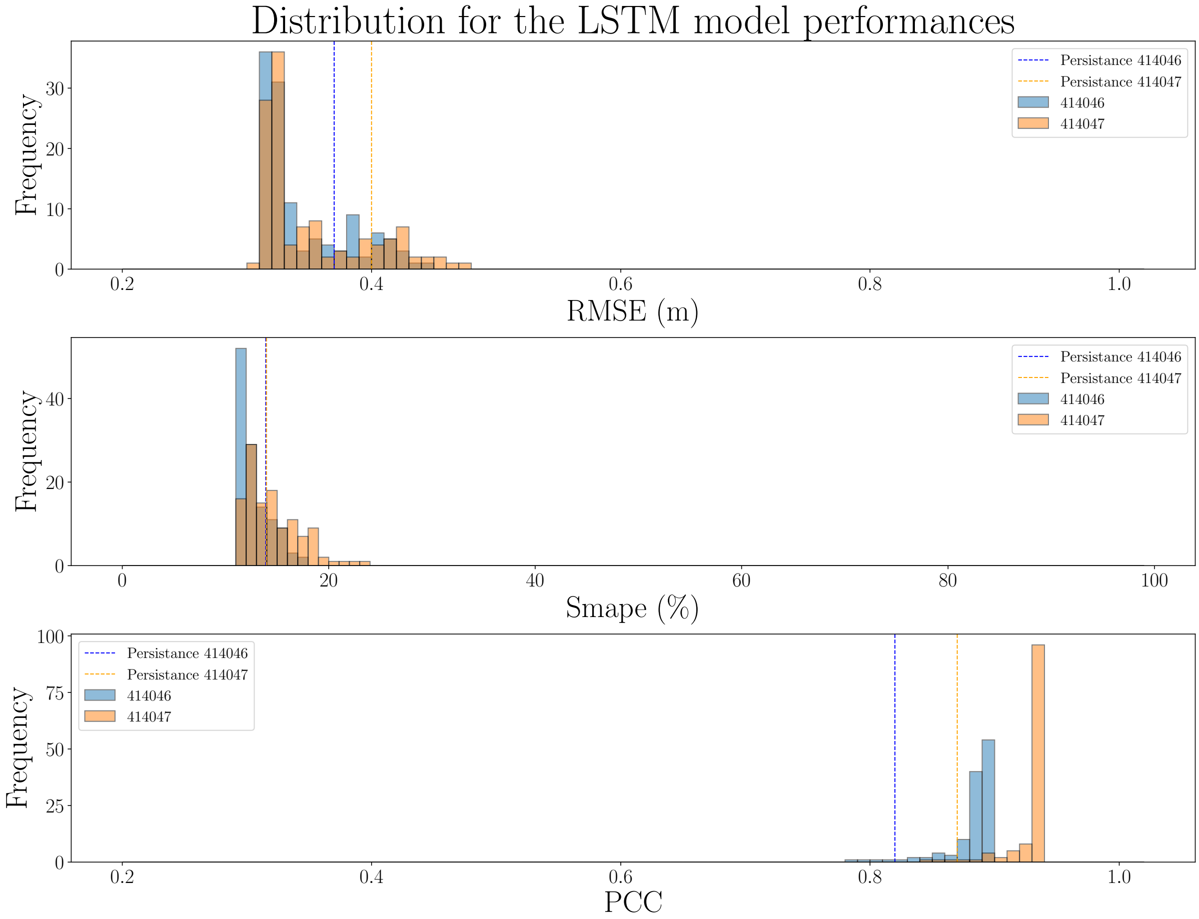


Figure C.4: The distribution of averaged performance over all time steps for the LSTM models using stepped input for buoy 414046 and 414047. The histograms use bin sizes of 0.01, 1 and 0.01 and the dashed line represents the accuracy of the persistence model. Note that smaller RMSE and sMAPE are desired, and a larger PCC.

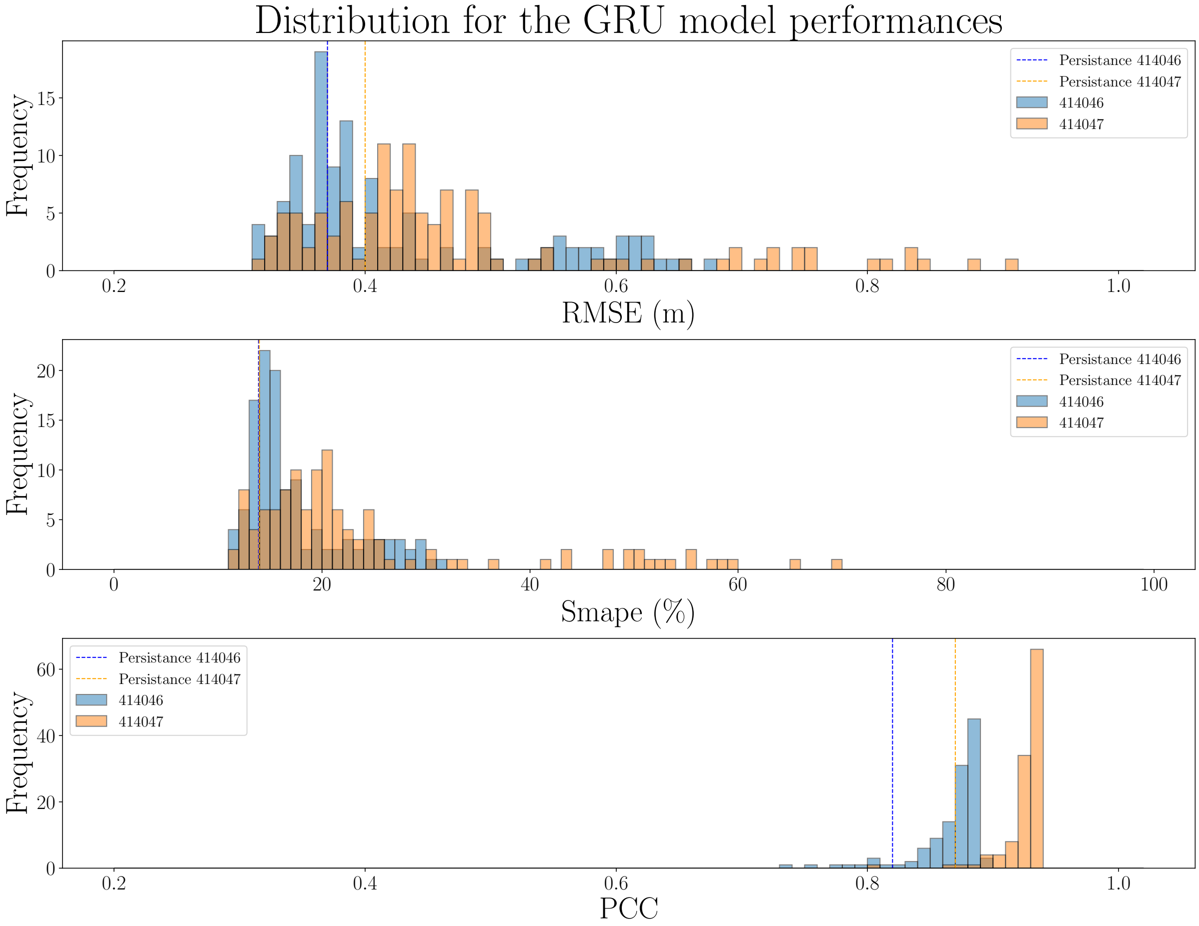


Figure C.5: The distribution of averaged performance over all time steps for the GRU models using stepped input for buoy 414046 and 414047. The histograms use bin sizes of 0.01, 1 and 0.01 and the dashed line represents the accuracy of the persistence model. Note that smaller RMSE and sMAPE are desired, and a larger PCC.

Appendix D

Decomposition plot

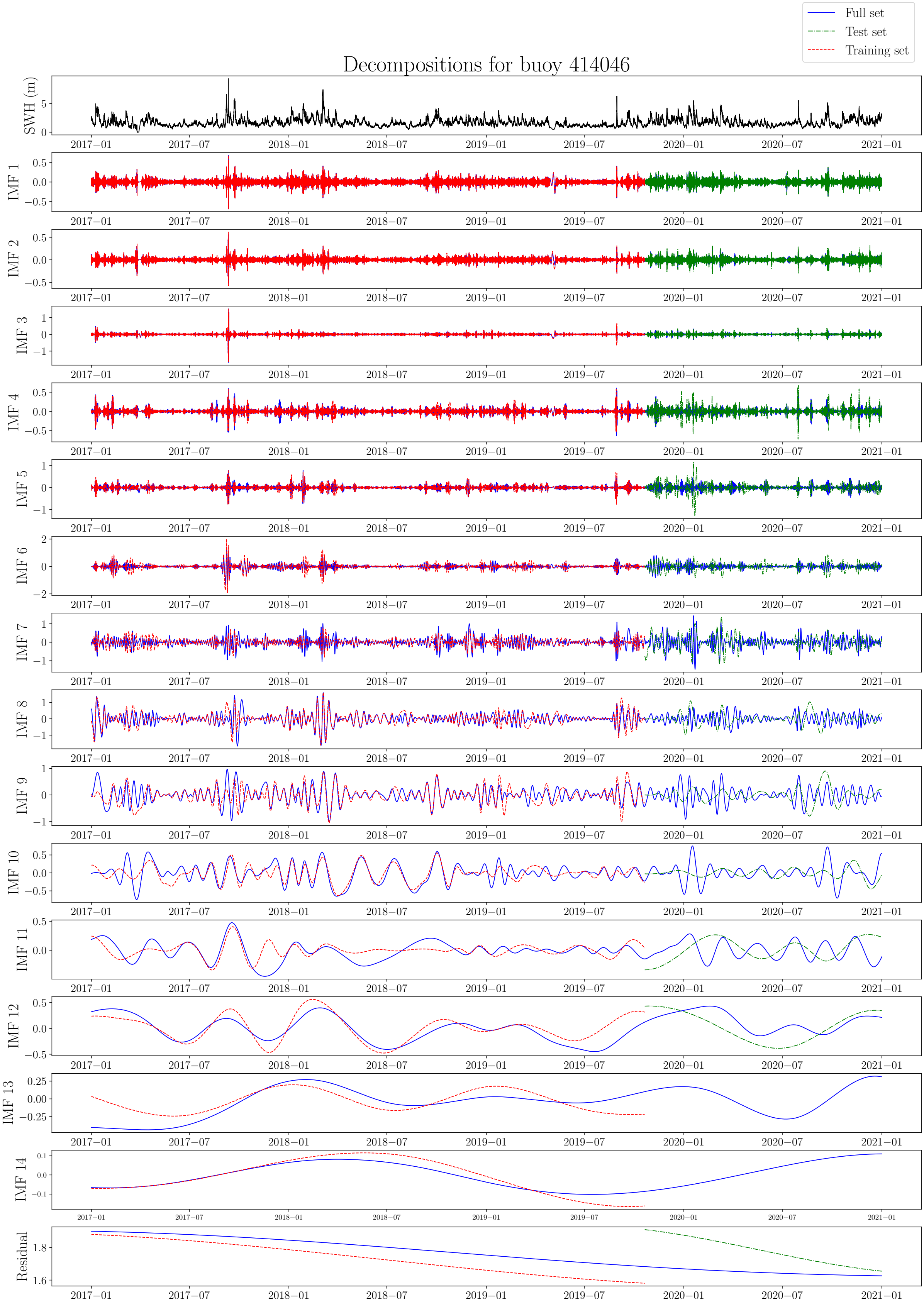


Figure D.1: EMD for the whole signal, a training set (70 %) and a test set (30 %) for buoy 414046 for the hourly SWH data. The black line represents the original signal, the blue line the decomposition of the whole set, the red line the decomposition of the training set and the green line the decomposition of the test set. Note that the training set only has 12 IMFs.

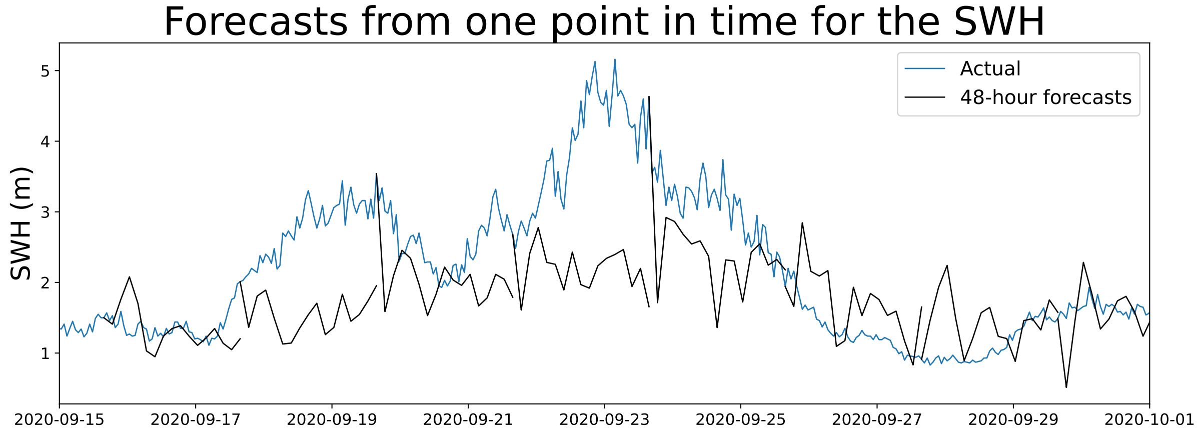


Figure D.2: Multiple 48-hour forecasts from a single point in time for buoy 414046 using EMD-framework one.

Bibliography

[1] NDBC - Measurement Descriptions and Units. URL https://www.ndbc.noaa.gov/measdes.shtml.

[2] D. A. Wing and M. C. Johnson. Ship operability prediction from long term directional wave records. *International Journal of Maritime Engineering*, 153(A2): 209–218, 2011. doi:10.5750/IJME.V153IA2.885.

[3] Rongyao Wang, Guoming Chen, Xiuquan Liu, Nan Zhang, and Wei Liu. Safety analysis of deep-sea mining pipeline deployment operations considering internal solitary waves. *Marine Georesources & Geotechnology*, 40(2):125–138, 2022. doi:10.1080/1064119X.2021.1889080.

[4] M. Hughes. Coastal waves, water levels, beach dynamics and climate change. *CoastAdapt, National Climate Change Adaptation Research Facility*, 8 2016.

[5] C H Lashley, S N Jonkman, J van der Meer, J D Bricker, and V Vuik. The influence of infragravity waves on the safety of coastal defences: a case study of the Dutch Wadden Sea. *Natural Hazards and Earth System Sciences*, 22(1):1–22, 2022. doi:10.5194/nhess-22-1-2022.

[6] T. W. Group. The WAM model—A third generation ocean wave prediction model. *Journal of Physical Oceanography*, 18(12):1775–1810, 1988. doi:10.1175/1520-0485(1988)018<1775:TWMTGO>2.0.CO;2.

[7] N. Booij, L. H. Holthuijsen, and R. C. Ris. The “SWAN" wave model for shallow water. *Proceedings of the Coastal Engineering Conference*, 1:668–676, 1997. doi:10.1061/9780784402429.053.

[8] M. C. Deo, A. Jha, A. S. Chaphekar, and K. Ravikant. Neural networks for wave forecasting. *Ocean Engineering*, 28(7):889–898, 2001. doi:10.1016/S0029-8018(00)00027-5.

[9] S. N. Londhe and V. Panchang. One-day wave forecasts based on artificial neural networks. *Journal of Atmospheric and Oceanic Technology*, 23(11):1593–1603, 2006. doi:10.1175/JTECH1932.1.

[10] S. Shamshirband, A. Mosavi, T. Rabczuk, N. Nabipour, and K. W. Chau. Prediction of significant wave height; comparison between nested grid numerical model, and machine learning models of artificial neural networks, extreme learning and support vector machines. *Engineering Applications of Computational Fluid Mechanics*, 14 (1):805–817, 2020. doi:10.1080/19942060.2020.1773932.

[11] A. Callens, D. Morichon, S. Abadie, M. Delpey, and B. Liquet. Using random forest and gradient boosting trees to improve wave forecast at a specific location. *Applied Ocean Research*, 104, 2020. doi:10.1016/J.APOR.2020.102339.

[12] R. Savitha and A. Al Mamun. Regional ocean wave height prediction using sequential learning neural networks. *Ocean Engineering*, 129:605–612, 2017. doi:10.1016/J.OCEANENG.2016.10.033.

[13] Google Maps. URL `https://www.google.com/maps`.

[14] C. P. Tsai, C. Lin, and J. N. Shen. Neural network for wave forecasting among multi-stations. *Ocean Engineering*, 29(13):1683–1695, 2002. doi:10.1016/S0029-8018(01)00112-3.

[15] C. E. Balas, L. Koç, and L. Balas. Predictions of missing wave data by recurrent neuronets. *Journal of waterway, port, coastal, and ocean engineering*, 130(5):256–265, 2004. doi:10.1061/(ASCE)0733-950X(2004)130:5(256).

[16] J. Mahjoobi, A. Etemad-Shahidi, and M. H. Kazeminezhad. Hindcasting of wave parameters using different soft computing methods. *Applied Ocean Research*, 30(1): 28–36, 2008. doi:10.1016/J.APOR.2008.03.002.

[17] I. Malekmohamadi, M. R. Bazargan-Lari, R. Kerachian, M. R. Nikoo, and M. Fallahnia. Evaluating the efficacy of SVMs, BNs, ANNs and ANFIS in wave height prediction. *Ocean Engineering*, 38(2-3):487–497, 2011. doi:10.1016/J.OCEANENG.2010.11.020.

[18] S. Fan, N. Xiao, and S. Dong. A novel model to predict significant wave height based on long short-term memory network. *Ocean Engineering*, 205(107298), 2020. doi:10.1016/J.OCEANENG.2020.107298.

[19] Martina Maria Pushpam P. and Felix Enigo V.S. Forecasting Significant Wave Height using RNN-LSTM Models. In *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*, pages 1141–1146, 2020. doi:10.1109/ICICCS48265.2020.9121040.

[20] F. C. Minuzzi and L. Farina. A deep learning approach to predict significant wave height using long short-term memory. 2022. doi:10.48550/arxiv.2201.00356.

[21] M. Li and K. Liu. Probabilistic prediction of significant wave height using dynamic Bayesian network and information flow. *Water*, 12(8):2075, 2020. doi:10.3390/W12082075.

[22] X. Tong, X. Xu, S. L. Huang, and L. Zheng. A Mathematical framework for quantifying transferability in multi-source transfer learning. *Advances in Neural Information Processing Systems*, 34:26103–26116, 2021. URL `https://proceedings.neurips.cc/paper/2021/hash/db9ad56c71619aeed9723314d1456037-Abstract.html`.

[23] R. Ye and Q. Dai. Implementing transfer learning across different datasets for time series forecasting. *Pattern Recognition*, 109:107617, 2021. doi:10.1016/J.PATCOG.2020.107617.

[24] Y. Wei and M. C. Chen. Forecasting the short-term metro passenger flow with empirical mode decomposition and neural networks. *Transportation Research Part C: Emerging Technologies*, 21(1):148–162, 2012. doi:10.1016/J.TRC.2011.06.009.

[25] C. F. Chen, M. C. Lai, and C. C. Yeh. Forecasting tourism demand based on empirical mode decomposition and neural network. *Knowledge-Based Systems*, 26: 281–287, 2012. doi:10.1016/J.KNOSYS.2011.09.002.

[26] S. Wang, N. Zhang, L. Wu, and Y. Wang. Wind speed forecasting based on the hybrid ensemble empirical mode decomposition and GA-BP neural network method. *Renewable Energy*, 94:629–636, 2016. doi:10.1016/J.RENENE.2016.03.103.

[27] M. D. Liu, L. Ding, and Y. L. Bai. Application of hybrid model based on empirical mode decomposition, novel recurrent neural networks and the ARIMA to wind speed prediction. *Energy Conversion and Management*, 233:113917, 2021. doi:10.1016/J.ENCONMAN.2021.113917.

[28] Y. Huang, S. Liu, and L. Yang. Wind Speed Forecasting Method Using EEMD and the Combination Forecasting Method Based on GPR and LSTM. *Sustainability*, 10 (10):3693, 2018. doi:10.3390/SU10103693.

[29] M. Ali and R. Prasad. Significant wave height forecasting via an extreme learning machine model integrated with improved complete ensemble empirical mode decomposition. *Renewable and Sustainable Energy Reviews*, 104:281–295, 2019. doi:10.1016/J.RSER.2019.01.014.

[30] S. Zhou, B. J. Bethel, W. Sun, Y. Zhao, W. Xie, and C. Dong. Improving significant wave height forecasts using a joint empirical mode decomposition–long short-term memory network. *Journal of Marine Science and Engineering*, 9(7):744, 2021. doi:10.3390/JMSE9070744.

[31] R. Alcorn. Wave Energy. *Future Energy: Improved, Sustainable and Clean Options for our Planet*, pages 357–382, 2013. doi:10.1016/B978-0-08-099424-6.00017-X.

[32] L. H. Holthuijsen. *Waves in oceanic and coastal waters*. Cambridge University Press, 2010. ISBN 9780511618536. doi:10.1017/CBO9780511618536.

[33] M. A. Tayfun. Narrow-band nonlinear sea waves. *Journal of Geophysical Research: Oceans*, 85(C3):1548–1552, 1980. doi:10.1029/JC085IC03P01548.

[34] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986. doi:10.1038/323533a0.

[35] B. Lim and S. Zohren. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A*, 379(2194), 2021. doi:10.1098/RSTA.2020.0209.

[36] L. Cornejo-Bueno, J. C. Nieto-Borge, P. García-Díaz, G. Rodríguez, and S. Salcedo-Sanz. Significant wave height and energy flux prediction for marine energy applications: A grouping genetic algorithm – Extreme Learning Machine approach. *Renewable Energy*, 97:380–389, 2016. doi:10.1016/J.RENENE.2016.05.094.

[37] J. Pearl. From Bayesian networks to causal networks. *Mathematical Models for Handling Partial Knowledge in Artificial Intelligence*, pages 157–182, 1995. doi:10.1007/978-1-4899-1424-8_9.

[38] K. P. Murphy. *Dynamic Bayesian networks: representation, inference and learning.* University of California, Berkeley, 2002. URL https://search.proquest.com/openview/3ef135ceb81b4b4e79c9a9ca649a622b/1?pq-origsite=gscholar&cbl=18750&diss=y.

[39] F. Yao. Machine learning with limited data. 2021. doi:10.48550/arxiv.2101.11461.

[40] I. Redko, A. Habrard, E. Morvant, M. Sebban, and Y. Bennani. Advances in domain adaption theory. *Advances in Domain Adaption Theory*, pages 1–208, 2019. doi:10.1016/C2016-0-05108-2.

[41] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2009. doi:10.1109/TKDE.2009.191.

[42] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P. A. Muller. Transfer learning for time series classification. *International conference on big data (Big Data)*, pages 1367–1376, 2018. doi:10.1109/BigData.2018.8621990.

[43] Y. Bao, Y. Li, S. L. Huang, L. Zhang, L. Zheng, A. Zamir, and L. Guibas. An information-theoretic approach to transferability in task transfer learning. *International Conference on Image Processing, ICIP*, pages 2309–2313, 2019. doi:10.1109/ICIP.2019.8803726.

[44] T. Wen and R. Keyes. Time series anomaly detection using convolutional neural networks and transfer learning. *CoRR*, 2019. doi:10.48550/arxiv.1905.13628.

[45] N. Laptev, J. Yu, and R Rajagopal. Reconstruction and regression loss for time-series transfer learning. *Proceedings of the Special Interest Group on Knowledge Discovery and Data Mining (SIGKDD) and the 4th Workshop on the Mining and LEarning from Time Series (MiLeTS)*, 2018. URL https://kdd-milets.github.io/milets2018/papers/milets18_paper_2.pdf.

[46] H. Xu, B. Xu, J. He, and J. Bi. Deep transfer learning based on LSTM model in stock price forecasting. *ACM International Conference Proceeding Series*, pages 73–80, 10 2021. doi:10.1145/3503181.3503194.

[47] P. Bloomfield. Fourier Analysis of Time Series: An Introduction. 2000. doi:10.1002/0471722235.

[48] C. Torrence and G. P. Compo. A Practical Guide to Wavelet Analysis. *Bulletin of the American Meteorological Society*, 79(1):61–78, 1 1998. doi:10.1175/1520-0477(1998)079.

[49] P. C. Deka and R. Prahlada. Discrete wavelet neural network approach in significant wave height forecasting for multistep lead time. *Ocean Engineering*, 43:32–42, 2012. doi:10.1016/J.OCEANENG.2012.01.017.

[50] A. K. Alexandridis and A. D. Zapranis. Wavelet neural networks: A practical guide. *Neural Networks*, 42:1–27, 2013. doi:10.1016/J.NEUNET.2013.01.008.

[51] R. Prahlada and P. C. Deka. Forecasting of time series significant wave height using wavelet decomposed neural network. *Aquatic Procedia*, 4:540–547, 2015. doi:10.1016/J.AQPRO.2015.02.070.

[52] N. E. Huang, Z. Shen, S. R. Long, M. C. Wu, H. H. Snin, Q. Zheng, N. C. Yen, C. C. Tung, and H. H. Liu. The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 454(1971): 903–995, 1998. doi:10.1098/RSPA.1998.0193.

[53] A. Zeiler, R. Faltermeier, I. R. Keck, A. M. Tomé, C. G. Puntonet, and E. W. Lang. Empirical mode decomposition - An introduction. *Proceedings of the International Joint Conference on Neural Networks*, 2010. doi:10.1109/IJCNN.2010.5596829.

[54] Z. Wu and N. E. Huang. Ensemble empirical mode decomposition: a noise-assisted data analysis method. *Advances in adaptive data analysis*, 1(1):1–41, 11 2009. doi:10.1142/S1793536909000047.

[55] M. E. Torres, M. A. Colominas, G. Schlotthauer, and P. Flandrin. A complete ensemble empirical mode decomposition with adaptive noise. *IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4144–4147, 2011. doi:10.1109/ICASSP.2011.5947265.

[56] J. Dong, W. Dai, L. Tang, and L. Yu. Why do EMD-based methods improve prediction? A multiscale complexity perspective. *Journal of Forecasting*, 38(7):714–731, 2019. doi:10.1002/FOR.2593.

[57] National Data Buoy Center. URL `https://www.ndbc.noaa.gov/`.

[58] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994. doi:10.1109/72.279181.

[59] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997. doi:10.1162/neco.1997.9.8.1735.

[60] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. 2014. doi:10.48550/arxiv.1412.3555.

[61] A. Graves, S. Fernández, and J. Schmidhuber. Bidirectional LSTM networks for improved phoneme classification and recognition. In *International conference on artificial neural networks*, volume 3697 LNCS, pages 799–804. Springer, Berlin, Heidelberg, 2005. ISBN 3540287558. doi:http://dx.doi.org/10.1007/11550907_163.

[62] T. Proietti and H. Lütkepohl. Does the Box–Cox transformation help in forecasting macroeconomic time series? *International Journal of Forecasting*, 29(1):88–99, 2013. doi:10.1016/J.IJFORECAST.2012.06.001.

[63] N. Gruber and A. Jockisch. Are GRU Cells More Specific and LSTM Cells More Sensitive in Motive Classification of Text? *Frontiers in Artificial Intelligence*, 3(40), 2020. doi:10.3389/FRAI.2020.00040/BIBTEX.

[64] S. Li, W. Li, C. Cook, C. Zhu, and Y. Gao. Independently Recurrent Neural Network (IndRNN): Building A Longer and Deeper RNN. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5457–5466, 2018. doi:10.48550/arxiv.1803.04831. URL `https://github.com/Sunnydreamrain/`.

[65] U. Khandelwal, H. He, P. Qi, and D. Jurafsky. Sharp nearby, fuzzy far away: how neural language models use context. *56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, 1:284–294, 2018. doi:10.48550/arxiv.1805.04623.

[66] P. Luo, X. Wang, W. Shao, and Z. Peng. Towards understanding regularization in batch normalization. 2018. doi:10.48550/arxiv.1809.00846.

[67] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32:8024–8035, 2019. URL `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`.

[68] G. Rilling, P. Flandrin, and P. Gonçalvès. On empirical mode decomposition and its algorithms. *IEEE-EURASIP workshop on nonlinear signal and image processing*, 3 (3):8–11, 2003. URL `https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.586.812&rep=rep1&type=pdf`.

[69] A. J Quinn, V. Lopes-dos Santos, D. Dupret, A. C. Nobre, and M. W. Woolrich. EMD: Empirical Mode Decomposition and Hilbert-Huang Spectral Analyses in Python. *Journal of Open Source Software*, 6(59):2977, 2021. doi:10.21105/joss.02977.

[70] T. Wang, M. Zhang, Q. Yu, and H. Zhang. Comparing the applications of EMD and EEMD on time–frequency analysis of seismic signal. *Journal of Applied Geophysics*, 83:29–34, 2012. doi:10.1016/J.JAPPGEO.2012.05.002.