ISE

Multivariate, Multistep Forecasting, Reconstruction And Feature Selection Of Ocean Waves Via Recurrent And Sequence-To-Sequence Networks

MOHAMMAD PIRHOOSHYARAN[1] AND LAWRENCE V. SNYDER[1]

[1]Industrial and Systems Engineering, Lehigh University

LEHIGH
UNIVERSITY®

Multivariate, Multistep Forecasting, Reconstruction and Feature Selection of Ocean Waves via Recurrent and Sequence-to-Sequence Networks

Mohammad Pirhooshyaran
Industrial & Systems Engineering
Lehigh University
mop216@lehigh.edu

Lawrence V. Snyder
Industrial & Systems Engineering
Lehigh University
lvs2@lehigh.edu

ABSTRACT

This article explores the concepts of ocean wave multivariate multistep forecasting, reconstruction and feature selection. We introduce recurrent neural network frameworks, integrated with Bayesian hyperparameter optimization and Elastic Net methods. We consider both short- and long-term forecasts and reconstruction, for significant wave height and output power of the ocean waves. Sequence-to-sequence neural networks are being developed for the first time to reconstruct the missing characteristics of ocean waves based on information from nearby wave sensors. Our results indicate that the Adam and AMSGrad optimization algorithms are the most robust ones to optimize the sequence-to-sequence network. For the case of significant wave height reconstruction, we compare the proposed methods with alternatives on a well-studied dataset. We show the superiority of the proposed methods considering several error metrics. We design a new case study based on measurement stations along the east coast of the United States and investigate the feature selection concept. Comparisons substantiate the benefit of utilizing Elastic Net. Moreover, case study results indicate that when the number of features is considerable, having deeper structures improves the performance.

1 Introduction

The intricate and ever-changing character of irregular waves necessitates the existence of a framework to estimate wave features in advance. Bringing the energy generated by ocean waves into the consumer's portfolios requires a consistent system capable of predicting ocean wave uncertainties. Most research on ocean wave prediction has focused on model-based approaches—for example, physics-based models that attempt to reproduce the frequency spectra of ocean waves and then to predict the resulting wave energy. However, these models are difficult to generalize.

Machine learning (ML) techniques allow model-free predictions of ocean wave characteristics to become practical and computationally efficient via monitoring of the huge amount of records taken in diverse bodies of water all around the world. We introduce recurrent networks integrated with Bayesian optimization and Elastic Net techniques to forecast future wave features and reconstruct missing data simultaneously.

There is a substantial literature on SWH forecasting; see Section 2 for a review. These papers investigate short- to long-term predictions of SWH at coastal and offshore sites with different water depths. Although there are marine applications solely based on SWH estimation, in most cases SWH is used as an input to evaluate other characteristics of an ocean wave, such as energy flux. Energy flux is the expected total sum of the wave energy available at a given location, and it plays a major role in designing marine structures from at least two broad perspectives. First, by modeling the energy flux of any potential location, one may identify remote locations where Wave Energy Converters (WEC) can be installed. Second, renewable energy of all sorts is well known to be less reliable sources with less resiliency against disruptions compared to energy produced by thermal power plants. Although wave energy is more predictable than wind or solar, it still suffers from randomness. Having a concrete framework to forecast significant wave height and ocean energy flux may directly improve the resiliency of electricity produced by this green source.

We propose a model that uses neural networks (NN) to forecast wave characteristics. Most studies that have used NNs to forecast or reconstruct wave characteristics use fully connected networks receiving inputs such as significant wave height (H_s) and/or average wave period (A) and outputs such as power ($\mathcal{P}$) and/or H_s. Recurrent Neural Networks (RNN), which contain natural temporal properties, have been considered in a few cases as well. The majority of these cases solely focus on predicting H_s. To the best of our knowledge, no studies have used sequence-to-sequence NNs to predict energy flux or reconstruct features. In this paper, we investigate the concept of multi-step-ahead univariate (H_s) and multivariate ($\mathcal{P}$) forecasting and feature reconstruction. Our contributions are as follows:

- We introduce two main networks. The first one is a simple combination of Long Short-Term Memory plus Fully Connected Layers at the end (LSTM+FCL). The LSTM part can be single- or multi-layered structures. The second network is a sequence-to-sequence (seqtoseq) [1] method containing two LSTMs as its encoder and decoder.

- We expand the framework to tackle the problem of reconstructing missing ocean wave data and feature selection based on information from their nearby buoys. We perform a detailed comparison between our models and the state-of-the-art papers on wave feature reconstruction [2, 3] and demonstrate that our approach outperforms those in the literature in terms of capturing missing H_s information.

- We modify an epoch-scheduled training scheme that is suitable for time series analysis in general as well as an elastic net concept, and we train the models based on these proposed schemes. A comparison between single- and multi-layered RNN, LSTM, and seqtoseq methods is conducted.

- We conduct Spearmint Bayesian optimization (GP-EI) to hypertune the models' parameters. Throughout the paper, we indicate the ith parameters that must be tuned as (*i*).

- We integrate elastic net concept into the model for the purpose of feature selection.

- We investigate the performance of four different optimization algorithms—SGD, RMSProp, Adam, and AMSGrad—for the seqtoseq network based on years of ocean wave data.

- We design a new dataset and a case study. The dataset is general and can be used for any feature selection and/or multivariate regression purposes. (Datasets are at the repository: NOAA-Refined-Stations)

The remainder of this paper is structured as follows: In Section 2, we discuss the necessary background on ocean waves, and we provide a concise literature review. Then, we introduce the LSTM+FCL and seqtoseq models and epoch-schedule training in section 3. Section 4 is dedicated to several comparisons, including long-term forecasts. The reconstruction framework is addressed in section 5. In section 6 we discuss about the feature selection with Elastic Net technique. The paper concludes in section 7.

2 Background and Literature Review

Statistical properties of the ocean surface suggest that for a given time and location, ocean waves may be viewed as the summation of a considerable amount of independent regular waves caused by wind sources or wave interactions [4]. Therefore, the ocean surface can be modeled as a zero-mean Gaussian stochastic process, considering all these waves [4, 5]. Buoys that contain wave-measurement sensors take a large number of samples from this process over time, and then, one may recover the spectral wave density $\mathcal{S}(w)$, which is the fast Fourier transform of the covariance matrix of the sea surface elevation [6–8].

Once $\mathcal{S}(w)$ of a surface wave has been estimated from the measurements collected, the Spectral Density Momentum (SDM) of order r is calculated as:

$$\text{SDM}(r) = \int_0^\infty w^r \mathcal{S}(w) dw. \tag{1}$$

By definition, we have $H_s = 4 \int_0^\infty \mathcal{S}(w) dw = 4\text{SDM}(0)^{\frac{1}{2}}$ and Average wave period $A = \frac{\text{SDM}(-1)}{\text{SDM}(0)}$. The energy flux $\mathcal{P}$ is then calculated as follows:

$$\mathcal{P} = \frac{\rho g^2}{64\pi} \int_0^\infty \frac{S(w)}{w} dw = \frac{\rho g^2}{64\pi}(H_s)^2 A, \tag{2}$$

where ρ is the ocean density, and g is the gravity constant.

The rest of this section contains three subsections. First, we provide a concise review of the literature about forecasting ocean wave characteristics. Second, we explore the novel work on oceanic wave feature reconstruction. Third, we discuss seqtoseq models from a machine learning point of view.

2.1 Ocean Characteristic Forecasting

Wave forecasting models can be categorized into model-based and model-free frameworks. Model-based approaches aim to use physical concepts such as climatic pressure, frictional dissipation and environmental interactions to find precise equations mimicking the behavior of wind waves. These methods are further classified based on their efforts to explicitly parameterize ocean wave interactions [9] into first, second and third generations. First-generation models try to construct the spectral wave structure solely based on the linear wave interactions. These models overestimate the power generated by water because of linear simplification and instead ignore any nonlinear transfer [10]. Second-generation models such as JONSWAP [11] examine coupled discrete spectral structures in such a way that the wave nonlinearities can be parameterized [12]. The most mature models are third-generation wave models such as WAM and simulating waves nearshore (SWAN) [13], which consider all possible generation, dissipation and nonlinear wave-wave interactions [10] along with current–wave interactions [9].

In contrast, data-driven, model-free approaches have become more popular recently with the advent of ML techniques. Fuzzy Systems (FS) [14–16], Evolutionary Algorithms (EA) [2, 3], Support Vector Machines (SVM) [17], and Deep Neural Networks [18–20] are so-called "soft computing techniques," which focus on the data structure in order to investigate possible relations and dependencies to forecast uncertain future events. Neural Networks are among the most powerful tools to approximate almost any nonlinear and complex functional relation. Recurrent Neural Networks (RNNs), a subclass of Neural Networks, exploit their internal memories to express temporal dynamic behavior, which makes them a suitable framework for forecasting complex systems. [21] explore the efficacy of several ML methods in terms of their accuracy in forecasting the significant wave height.

The general goal of any forecasting method is to find an accurate short- or long-term forecast of the variable under study. (See [22, 23] for new forecasting studies.) However, there exist ML approaches particularly designed for forecasting of wave characteristics. For more thorough discussion of the literature on ocean power and significant wave height forecasting one may refer to past reviews [24–30]. [31] implements a Nonlinear Autoregressive (NAR) neural network to forecast exponentially smoothed ocean wave power using Irish Marine Institute data. A Minimal Resource Allocation Network (MRAN) and a Growing and Pruning Radial Basis Function (GAP-RBF) network are implemented and tested on three geographical locations by [20]. The significance of a node in the GAP-RBF network is measured as its contribution to the network output, and the node is added or pruned accordingly. Cascade-forward and feed-forward neural networks are implemented in [32] to predict the wave power itself in the absence of spectral wave data. An ensemble of Extreme Learning Machines (ELM) is presented in [33] to predict the daily wave height. The authors use the previous hours' of wave heights along with features such as air to sea temperature difference, atmospheric pressure, wind speed to predict the next 6-hour wave height. Two computationally efficient supervised ML approaches are introduced by [34] and compared with the SWAN model [35]. An integrated numerical and ANN approach is introduced by [36] to predict waves 24 hours in advance at different buoys along the Indian Coastline. Nonlinear and non-stationary H_ss are studied in [37] based on integrated Empirical Model Decomposition Support Vector Regression (EMD-SVR). Forecasting of extreme events such as hurricanes is examined by Dixit et al. [38] via a Neuro Wavelet Technique (NWT). A recent work [39] aims at predicting H_s 48 hours into the future utilizing a hybrid model combining neural network with wavelets (WLNN). As mentioned before non of these work take into consideration sequence-to-sequence networks to investigate its short to long term forecasting performance yet alone its combination with Spearmint Bayesian optimization and elastic net technique.

2.2 Reconstruction of Ocean Characteristics

In contrast to forecasting frameworks, in which the goal is typically to estimate wave features such as H_s and $\mathcal{P}$ in the future based on historical data, reconstruction models aim to use available information about wave features to reconstruct H_s, $\mathcal{P}$, or other (usually missing) features. Here, the assumption is that the model has access to up-to-date measurements of the wave features, except for the one(s) they want to reconstruct. This is commonly due to missing measurement data. Therefore, the prediction continues one step ahead (or a few steps ahead) into the future. Hence, the aim is to tackle the problem of extracting the ocean wave information of a location purely based on other available features. The framework is useful for estimating the missing data of a station using the knowledge obtained from its neighbors but any available information of the same station can be utilized as well.

The paper [2] and its subsequent improvement [3] are dedicated to reconstructing ocean waves based on Evolutionary Algorithms (EA). The authors address the problem primarily via a Grouping Genetic Algorithm (GGA) and Bayesian optimization Grouping Genetic Algorithm (BO-GGA). They utilize GGA and/or BO-GGA to select the wave features of nearby stations suitable to predict the desired wave feature of the location with missing data, and then obtain their predictions via simple ELM or SVM. The paper reconstructs the significant wave height of the NOAA buoy 46069 for the year 2010 solely based upon the information provided from two adjacent buoys.

3

2.3 Sequence-to-Sequence Neural Networks

Deep Neural Networks (DNN) are among the most successful tools for classification and regression. DNNs achieve state-of-the-art performance in many applications. Although a simple feed-forward DNN can be applied in many systems, they require the system to have fixed input and output size. Recurrent Neural Networks such as Long-Short-Term Memory (LSTM) networks tackle this limitation in the sense that they do not need a fixed input size [40]. LSTMs can observe an input sequence of arbitrary size sequentially (one time step at a time) to provide the rest of the network with a large, fixed-sized vector representing the input [41, 42]. Furthermore, LSTMs remember long-range feature propagation based on a sigmoid layer called "forget gate." Seqtoseq models use two separate recurrent structures. They vary from basic recurrent networks in the sense that the network fully reads the input sequences before it generates any outputs. The first network is usually an LSTM which reads (encodes) the input of any size and maps them to a fixed-sized output. The second structure generally receives the fixed-sized output vector of the first LSTM and maps (decodes) them to a desirable output space [1]. Having encoding and decoding as separate steps gives the model flexibility and stability when handling complex sequence structures [43].

An important technique for training recurrent networks is Teacher Forcing (TF), which forces the network to observe the previous ground truth output instead of the one it already predicted. In other words, TF, keeping the network complex structure, converts any long-term prediction structures to one-step-ahead forecasts. This can greatly increase the network's ability to learn, thereby reducing its learning time [44, 45]. On the other hand, one can argue that the new model is not solving the same problem anymore. Therefore, there often exists a large gap between the testing error and training error for the models trained by TF; that is, the model encounters a major overfitting problem. Hence, [46] introduces a modification of scheduled training that captures the benefit TF while avoiding overfitting. Scheduled training is a soft technique. That is, it starts with a TF scheme and, after the model passes the warm-up stage and the network's weights have gone in the correct update direction, it alters the original network with a specified scheme. Therefore, the model enjoys stability and the probability of overfitting decreases. In our work, we revisit scheduled training and introduce epoch-scheduled training for forecasting.

3 The Models

3.1 Overview

In this section, we discuss the proposed models. We denote an input sequence to the model as $(x_1, \cdots, x_T)$ and its output sequence as $(y_1, \cdots, y_{T'})$, where T and T' need not be equal. In the case of predicting $\mathcal{P}$, we have $x_t = [H_{s_t}, A_t]$ and $y_{t'} = [\mathcal{P}_{t'}]$. The purpose of the models is to calculate the conditional distribution $p(y_1, \cdots, y_{T'} | x_1, \cdots, x_T)$. T' indicates how far into the future the forecast should go. For example, for hourly-resolution data, if the model is a day-ahead energy flux forecast, then $T' = 24$. T, on the other hand, is a model parameter and must to be tuned (*1*).[1] T can be interpreted as the number of recurrent loops in the structure. Seqtoseq structures contain two independent recurrent networks: encoder and decoder. We use the LSTM network as an encoder, consisting of input i, cell state s, output o and forget f gates (nodes) [47]. The standard LSTM equations [40, 41] for time-step t are as follows :

$$f_t = \text{sigm}\left(W_{fx}x_t + W_{fh}h_{t-1}\right) \tag{3}$$
$$i_t = \text{sigm}\left(W_{ix}x_t + W_{ih}h_{t-1}\right) \tag{4}$$
$$o_t = \text{sigm}\left(W_{ox}x_t + W_{oh}h_{t-1}\right) \tag{5}$$
$$\tilde{s}_t = \tanh\left(W_{\tilde{s}x}x_t + W_{\tilde{s}h}h_{t-1}\right) \tag{6}$$
$$s_t = f_t \odot s_{t-1} + i_t \odot \tilde{s}_t \tag{7}$$
$$h_t = o_t \odot \tanh(s_t), \tag{8}$$

where $x_t \in \mathbb{R}^d$, $h_t \in \mathbb{R}^h$, $f_t \in \mathbb{R}^h$, $i_t \in \mathbb{R}^h$, $o_t \in \mathbb{R}^h$, $s_t \in \mathbb{R}^h$ and $\tilde{s}_t \in \mathbb{R}^h$ for all $t = 1, .., T$ are the input, hidden state, forget activation, input activation, output activation, cell state and auxiliary cell state vectors, respectively, for the LSTM network. W_{ij} is the weight matrices corresponding to the dimensions of the gate vectors i and j. The sigmoid function is given by $\text{sigm}(x) = \frac{1}{1+e^{(-x)}}$, $\forall x \in \mathbb{R}$; its return value is monotonically increasing in the open interval $(0, 1)$. The notation $\odot$ indicates element-wise (Hadamard) matrix product, which exists only if the matrix dimensions are the same.

The weight matrices can be initialized randomly with Gaussian distribution. Further, h_0 and s_0 are initialized by zero vectors. In energy flux forecasting, $d = 2$, and h refers to the hidden vector size, which needs to be tuned (*2*). The biases are omitted in the formulas because one may incorporate them easily through the weight matrices by adding one extra element to each vector mentioned.

---

[1] Recall that we use *i* to denote the ith tunable parameter.

A forget gate responds to the question, "which data should the model keep from the previous cell state vector?" In (3), the sigmoid function outputs a number between 0 and 1, which indicates the importance of the previous plus current inputs. This value is directly multiplied with the cell state vector of the previous cell state in the first term of (6). The input gate (4) decides which new information should be collected. The second term of (6) updates this new information. The updated cell state vector is ready to go through the next part of the network. Furthermore, the hidden state is merged into the cell state through (8).

The purpose of the encoder is to find a representation of the $(x_1, \cdots, x_T)$ sequence as a fixed-sized vector v. For the "LSTM+FCL" network, we create a single fully connected layer that receives a fixed-sized vector v as its input and outputs $y_{t'} = [\mathcal{P}_{t'}]$.

For the seqtoseq network, we have another recurrent network as the decoder. The decoder considers vector v as its input and the last encoder hidden state (h_T) as its initial hidden state $h_{t'=0}$ and runs a similar recurrent construction to decode the output sequence. We omit the equations, as they are similar to those above. Furthermore, we let the decoder update its weight parameters independently. In other words, the encoder and decoder do not share parameters. This doubles the number of parameters to be tuned and multiplies the training time by a constant, but we allow this in the hope of achieving better performance. The decoder operates a static recurrent structure rather than a dynamic one. That is, the decoder creates an unrolled computational graph of fixed length due to the fixed-sized input vector v. In addition, both encoder and decoder can be deep structures. That is, the number of stacked LSTM layers (*3*) for each of them can vary.

The seqtoseq model attempts to find the output sequence distribution as follows:

$$p(y_1, \cdots, y'_T | x_1, \cdots, x_T; W) = p(y_1, \cdots, y'_T | v; W) \tag{9}$$

$$= p(y_1 | v; W) \prod_{t'=1}^{T'} p(y_{t'+1} | v, \mathbb{L}(y_1, \cdots, y_{t'}); W), \tag{10}$$

where W represents all the model weights to be tuned and $\mathbb{L}$ is a binary indicator representing whether the model sees the actual previous measurements or their predictions. The first equality (9) relies on the encoder and emphasizes once again that the model encodes the whole input $(x_1, \cdots, x_T)$ into the vector v before starting to decode and drops the input thereafter. The concept of multiplying conditional probabilities in the second equality (10) comes from the recurrent structure of the decoder. In other words, based on the trained weights and the vector v, the model tries to forecast the first token (element) of the output sequence. In general, at time step $t' + 1$, the model has access to $\mathbb{L}(y_1, \cdots, y_{t'})$ of the output sequence. In the TF strategy [44, 45], the model has access to the actual true outputs at training time (i.e., $\mathbb{L}(y_1, \cdots, y_{t'}) = (y_1, \cdots, y_{t'})$) and then to the predicted ones (i.e. $\mathbb{L}(y_1, \cdots, y_{t'}) = (\hat{y}_1, \cdots, \hat{y}_{t'})$) at testing time. In the scheduled strategy, however, one may flip a coin [46] with probability ϵ'_t for all $t' = 1, \cdots, T' - 1$ to decide whether to use the actual output (with probability ϵ'_t) or its prediction by the model itself (with probability $1 - \epsilon'_t$). For instance, the probability of having all true outputs $(y_1, \cdots, y_{t'})$ during training is $\epsilon_1 \epsilon_2 \cdots \epsilon_{T'}$. This scheme has been introduced for Machine Translation (MT) tasks, where the number of possible output tokens is as large as the dictionary size. Furthermore, in the MT framework the model faces embedded input and sequence. In contrast, in a forecasting framework, each output token belongs to $\mathbb{R}$. Therefore, using $\epsilon_{t'}$ for each token t' may result in a combination of true outputs along with predicted ones, which is not particularly useful. Instead, we flip a coin at the beginning of each epoch and stick to the plan for the entire epoch. So, from now on we denote the sequence as $\{\epsilon_{ep}\}$, where ep is the training epoch number. Intuitively, the sequence $\{\epsilon_{ep}\}$ should be decreasing, which encourages the model to use the predicted output towards the end of the training. [46] introduces Linear, Exponential and Inverse Sigmoid Decay sequences. We modify the Inverse Sigmoid Decay to use for epoch-scheduled training as follows:

$$\epsilon_{ep} = \frac{k}{k + e^{(ep/k)}},$$

where k is a parameter to tune (*4*). Increasing k increase the probability of receiving true values. For example, for a training scheme of size 20 epochs, $k = 20$ means that we receive the true outputs with at least 0.86 probability. Hence, in the tuning process, we compare the magnitude of k with the number of epochs.[2] Figure 1 illustrates a schematic of the epoch-scheduled training sequence to sequence network. An expanded view of a single LSTM cell structure is presented in Figure 2, with details and equations (3)-(8) marked in the figure. Hidden and state cells propagate through the encoder and decoder networks. In Figure 1, ϵ_{ep} expresses the probability that a given decoder cell sees the actual outputs, where ep index iterates over all epochs.

---

[2]However, we manually force the model to use the predicted values for the last two epochs.
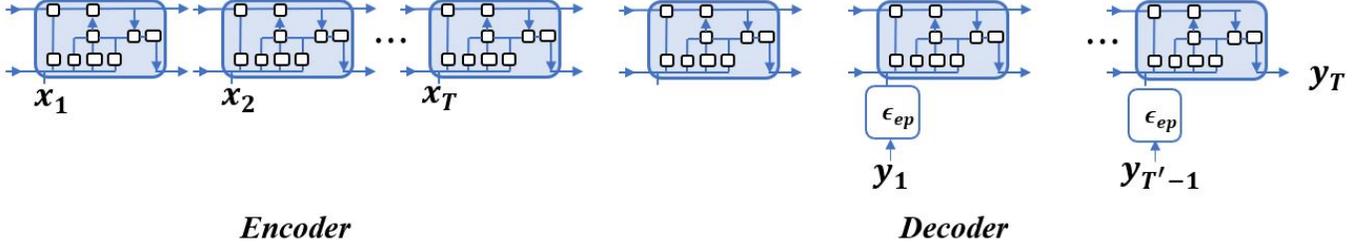
Figure 1: Schematic diagram of the epoch-scheduled training sequence to sequence network consisting of the encoder and decoder parts.
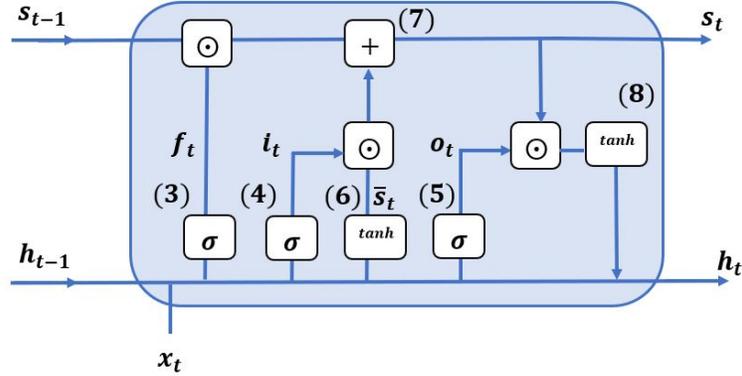

Figure 2: LSTM cell: σ and $tanh$ are sigmoid and tangent hyperbolic activation functions, $\odot$ indicates element-wise (Hadamard) matrix product, hidden and state cells propagate through the network.

3.2 Loss Function & Optimizer

We use Mean Squared Error (MSE) plus a 2-norm regularizer as the loss function for the whole network:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(y^i - \hat{y}^i\right)^2, \tag{11}$$

where n is the size of the measurements and $\hat{y}^i$ is a vector of size T' representing the model's prediction of y^i, which is the actual power outputs. The data, however, is chopped up and fed via mini batches, the size of which is tunable (*5*) as a power of 2, which is a common practice in optimization algorithms. Regardless of the algorithm used, MSE is clearly convex with respect to $\hat{y}^i$. But we do not have direct control over $\hat{y}^i$. Instead, $\hat{y}^i = N(x^i; W)$ where N is the seqtoseq network structure and x^i is the i-th network input. The loss function is generally non-convex with respect to W, which justifies our use of a 2-norm regularizer [48, 49]. Hence, a term $\lambda\|w\|_2^2$ is added to the loss function, where w is a vector that stacks all the trainable parameters within the weight matrices and λ (*6*) is a regularization parameter to be tuned. Therefore, the loss function equals

$$f_w(x) = \frac{1}{n}\sum_{i=1}^{n}\left(y^i - N(x^i; w)\right)^2 + \lambda\|w\|_2^2.$$

Through backpropagation, we calculate the gradient of the loss function and update the weights via a first-order optimization algorithm. Stochastic Gradient Descent (SGD) directly updates w in the direction of the negative of the mini-batches' gradient in an iterative manner. That is, if b_i is the ith mini-batch, then we have

$$w_{i+1} := w_i - \alpha_i \nabla_w \bar{f}_w(x),$$

where

$$\bar{f}_w(x) = \frac{1}{|b_i|}\sum_{b_i}\left(y^i - N(x^i; w)\right)^2 + \lambda\|w\|_2^2$$

and α_i is the learning rate (*7*) at step i, which needs to be tuned. There exist several modifications of SGD, such as RMSProp. Adaptive moment estimation (Adam) [50] has been successfully implemented as another first-order method. In Adam, the model's weight is updated as follows:

6

$$w_{i+1} = w_i - \frac{\alpha_i}{\sqrt{\hat{v}_i} + \epsilon}\hat{m}_i, \tag{12}$$

where

$$\hat{m}_i = \frac{m_i}{1 - \beta_1^i} \tag{13}$$

$$\hat{v}_i = \frac{v_i}{1 - \beta_2^i} \tag{14}$$

$$m_i := \beta_1 m_{i-1} + (1 - \beta_1)\nabla_w \bar{f}_w(x) \tag{15}$$

$$v_i := \beta_2 m_{i-1} + (1 - \beta_2)\nabla_w^2 \bar{f}_w(x), \tag{16}$$

β_1 and β_2 are momentum-like parameters, and ϵ serves to reduce numerical issues. Adam aims to update each element of w with respect to its size (adaptively); it stores only an exponentially decaying average of past squared gradients. The authors of [50] propose $\beta_1 = 0.9$, $\beta_2 = 0.99$, and $\epsilon = 1e-8$ to be efficient. In some cases, such as machine translation, in which we are dealing with RNN structures, similar to our forecasting context, Adam suffers from using exponential averaging over the past squared gradients [51]. AMSGrad has been proposed to tackle this issue by simply using the maximum of the past squared gradients. In other words, we update $\hat{v}_i = \max\{\hat{v}_{i-1}, v_i\}$. Below, we first compare the efficiency of the SGD, RMSProp, Adam, and AMSGrad algorithms, and then optimize the energy flux forecasting model mostly with Adam and AMSGrad.

3.3 Data

Satellite altimeters and buoy measurements are the two most common sources of data for wave feature forecasting [52]. In this study, we primarily use buoy measurements from the National Oceanic and Atmospheric Administration (NOAA). Each buoy provides measurements of significant wave height, wind speed, wind direction, average wave period (A), sea level pressure, gust speed, air temperature, and sea surface temperature, at resolutions of 10 minutes to 1 hour.

The National Data Buoy Center (NDBC) maintains three major data sets, consisting of moored buoys, drifting buoys, and Coastal-Marine Automated Network (C-MAN) stations, which are located alongside U.S. coastal structures. Along with other air and water features, they monitor wave energy spectra, from which H_s and A can be obtained via equation (1). Each site has an identifier (ID). IDs are in the form of five digits, except for C-MAN stations, which have alphanumeric IDs. The first two digits are assigned to a continental region, and the last three indicate a specific location. For instance 41001, 41002 and 41004 are Atlantic Ocean sites near the southeastern U.S. The sites typically have hourly resolution, which produces 8760 H_s and A data points per year. The data sets use 99.00 to indicate missing measurements. Some sites, however, aimed to report 10-minute-resolution data in 2017. For instance, among the above-mentioned stations, 41002 continues to provide hourly data, while 41001 and 41004 try to provide 10-minute-resolution data, but for the most part, their data is still hourly, with the non-hourly values filled by 99.00. Quite a few refined stations have been collected for this study. We consider a buoy to be a "refined station" when it is active and has a year with at least 1000 meaningful data points (approximately 11.4%). We calculate energy flux $\mathcal{P}$ via equation (2) for the datasets that remain after this process. Table 1 displays the buoys investigated in this research, along with some information about them. The locations are chosen from nearshore South Pacific Ocean, the Gulf of Mexico, and the North Atlantic Ocean, where the water depths range from approximately 100 m to 5 km. We divided the data into 3 parts: 60% for training, 20% for hyperparameter tuning and 20% for testing unless we are comparing with alternative approaches and they used another division. The model has no access to the testing data in any manner. Table 1 indicates the training and testing data sizes as well. The validation (tuning) is similar to testing hence we did not show that.

We argue that some of the studies in the literature are insufficient from a data point of view in two broad senses. First, for papers that use real buoy measurements, there is usually extensive data manipulation and preprocessing. Sometimes the proposed models see only subsamples of measurements in order to produce smaller forecasting metric errors. For example, [53, 54] choose subsample of 50 daily points out of 8 months of data (Jan 1–Aug 30, 2015) and report Root Mean Squared Error (RMSE) as an error measurement metric. Not only is this metric scale-dependent and can be changed by shrinking the data size, but this selection of points can break the temporal notion of the data as well. Therefore it has a direct effect on the nonlinear functional relation that we aim to predict. Moreover, [31] produce the data using the power matrix method. Hence, the accuracy of the method heavily depends on the accuracy of the data-producing procedure. In our study, after selecting a buoy, we do not conduct *any* data manipulation or preprocessing, other than cleaning not-a-number (NaN) values from the NOAA data. Otherwise, the input to the model is the raw data collected from NOAA. (See https://www.ndbc.noaa.gov/wave.shtml to see how spectral wave data are derived from buoy motion measurements.)

Table 1: Information of selected refined buoys

Usage	Station ID	Lon (W)	Lat (N)	Water Depth (m)	Years Interval	Training Points #	Test Points #
Forecasting:	41043	64.830	21.124	5271	[2007,2017]	52911	17904
	41040	53.045	14.554	5112	[2005,2017]	54794	18617
	42056	84.938	19.918	4526	[2005,2017]	58206	19548
	32012	85.078	19.425	4534	[2007,2017]	49280	16428
	41049	62.938	27.490	5459	[2009,2017]	44816	14996
	41060	51.017	14.824	5021	[2012,2017]	24817	8272
	41001	72.617	34.625	4453	$[1976,2017]\backslash\{1979,2008\}$[1]	134018	44791
	51002	157.696	17.037	4934	$[1984,2017]\backslash\{2013\}$[1]	132560	44507
	46086	118.036	32.507	1838	[2003,2017]	69044	23233
	46013	123.307	38.238	122.5	[1981,2017]	164022	54980
	41048	69.590	31.860	5340	[2007,2017]	49820	16705
	46084	136.102	56.622	1158	$[2002,2017]\backslash\{2009\}$[1]	64186	21606
	46083	137.997	58.300	136	$[2001,2017]\backslash\{2014\}$[1]	58440	18273
Reconstruction:	46042	122.398	36.785	1645.9	[2009,2010]	4687	4576
	46025	119.049	33.761	888	[2009,2010]	4687	4576
	46069	120.229	33.663	986	[2009,2010]	4687	4576
Feature Selection:	44007	70.141	43.525	26.5	[2018]	2737	685
	44008	69.248	40.504	74.7	[2018]	2737	685
	44009	74.702	38.457	30	[2018]	2737	685
	44013	70.651	42.346	64	[2018]	2737	685
	44014	74.840	36.606	47	[2018]	2737	685
	44017	72.049	40.693	48	[2018]	2737	685

1: These years are excluded based on unsatisfactory amount of available data

3.4 Error Metrics

We use the following forecasting Error Metrics:

- Root Mean Square Error (RMSE) is one of the most commonly used regression error metrics. It equals the square root of the MSE, given in (11), and it provides an useful measure of forecasting quality. The closer the RMSE gets to 0, the better the fit the prediction gives. In general, it is difficult to choose an appropriate threshold such that the prediction is deemed "accurate" if the RMSE is less than that threshold, because because the RMSE is scale dependent and not robust to outliers.

- HUBER loss, given two sets of points y and $\hat{y}$, is defined as

$$H_\delta = \frac{1}{n}\sum_{i=1}^{n}\mathcal{L}_\delta\left(y^i, \hat{y}^i\right), \tag{17}$$

  where

$$\mathcal{L}_\delta\left(y^i, \hat{y}^i\right) = \begin{cases} \frac{1}{2}(y^i - \hat{y}^i)^2 & \text{for } |y^i - \hat{y}^i| \leq \delta \\ \delta|y^i - \hat{y}^i| - \frac{1}{2}\delta^2 & \text{otherwise.} \end{cases} \tag{18}$$

  Generally, $\delta = 1$ is an acceptable threshold. HUBER loss is scale dependent but is less sensitive to outliers than RMSE is.

- Pearson Correlation Coefficient (CC) is defined as:

$$\rho_{y,\hat{y}} = \frac{\text{cov}(y, \hat{y})}{\sigma_y \sigma_{\hat{y}}}, \tag{19}$$

  where $\text{cov}(\cdot)$ is covariance. We have $-1 \leq CC \leq 1$. Although CC captures linear similarities of its inputs very well, several characteristics of nonlinear relations are ignored.

- Mean Arctangent Absolute Percentage Error (MAAPE), given two sets of points y and $\hat{y}$, is defined as:

$$\frac{1}{n}\sum_{i=1}^{n} arct\left(|\frac{y^i - \hat{y}^i}{y^i}|\right). \tag{20}$$

  MAAPE is scale-independent and, unlike MAPE, overcomes problematic cases as y^i goes to zero for all $i = 1, \cdots, n$.

3.5 Hyperparameter Tuning

In previous sections we identified tunable parameters $(^*i^*)$, $i = 1, 2, ..., 7$. Here, we use Spearmint Bayesian optimization [55] as a tuning tool. The algorithm is capable of handling integer parameters, as in our case [56]. The algorithm treats the seqtoseq forecasting model as a random black box function and places a Gaussian Process (GP) prior over it. After collecting function evaluations, it extracts posterior information based on the Expected Improvement (EI) observed. Hence, GP-EI-OptChooser with Monte Carlo approximation is selected, which tells Spearmint which of the candidates to execute next. The values of each parameter that we consider are listed in Table 3.5.

Table 2: Possible values for parameters of the sequence-to-sequence model.

hidden size	$\{32,64\}$
time step	$\{10,20,30,40,50,60\}$
batch size	$\{16,32,64,128,256\}$
learning rate α_i	$\{0.001,0.002,0.003,0.004,0.005,0.006,0.007,0.008,0.009,0.010\}$
# stacked layers	$\{1,2,4\}$
2-norm regularizer λ	$\{0.001,0.002,0.003,0.004,0.005,0.006,0.007,0.008,0.009,0.010\}$
k scheduled training	$\{0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0\} \times$ epochs

Spearmint only has access to the validation dataset and the model that has already been trained on the training dataset. Any hyperparameter tuning algorithm such as Spearmint can have its own objective function. That is, Spearmint's objective function during validation can be separate from the seqtoseq model's during training (which, recall, is MSE plus a 2-norm regularizer). That is, the model enjoys (MSE plus 2-norm regularization) convergence properties in training while enjoys using MAAPE scale independency and upper bound $(\pi/2)$ during the tuning process. Figure 3 illustrates the process of using Spearmint for hyperparameter tuning. We call the number of function evaluations as the "algorithm budget."

4 Experimental Results

In this section, we first report on the model's performance utilizing several optimization algorithms and compare them to those of standard Neural Networks on buoy measurements. Next, we demonstrate the performance of our model for long-term forecasting (48 steps ahead) of H_s. Finally, we tackle the problem of constructing wave characteristics such as H_s for unknown locations based on information from known locations. We wrote the code for our method and its alternatives in Python 3.6 using the TensorFlow package (running on a GeForce 1050 GTX GPU), except the code for hyperparameter tuning via Spearmint, whose most stable version for our framework is written in Python 2.7.

4.1 Comparison of Optimization Algorithms

In this section, we aim to compare the effectiveness of the SGD, RMSProp, Adam, and AMSGrad optimization schemes based on test error for four of the refined buoys. Figure 4 displays the algorithm test errors versus the number of training epochs, considering all five error metrics introduced earlier. For the cases of RMSE and HUBER losses, the values on the y-axis do not carry a clear meaning. In other words, we only hope to reach the smallest possible values for these two loss functions, but the actual values do not tell us much. One can observe that, even at the end of the training epochs, RMSE and HUBER losses are not negligible. Note that the error we experience during the calculation of $\mathcal{P}$ is of the third power of the error we encounter forecasting H_s. A similar difference has been pointed out by [3]. Moreover, RMSE and HUBER are both scale-dependent, and the test sets include years of data, which may result in errors piling up over time. In contrast, MAPE and MAAPE are scale independent, and therefore their values have more intuitive meaning. In addition, their behavior is similar to each other, considering that MAAPE values are consistently lower than those of MAPE. Therefore, we mainly focus on MAPE and MAAPE for subsequent experiments. SGD and, in a few cases, AMSGrad require more epochs to reach their best test accuracy, while Adam usually reaches its peak faster. RMSProp and Adam perform similarly, with Adam experiencing a marginal lead. AMSGrad and Adam exhibit excellent performance in terms of RMSE error. We used a learning rate of $\alpha = 0.001$ with a decay of 0.9 for all the algorithms. This is to make the algorithms more stable; otherwise, having a fixed learning rate would cause most algorithms, especially SGD, to encounter drastic changes towards the end of the optimization process. HUBER losses are the most challenging loss functions to deal with. One can see sizable fluctuations, even near the end of the training, for stations 41049 and 32012. SGD shows inferiority in terms of test error; however, based on its simple update, it enjoys the best results in terms of time needed for training the algorithms. SGD is especially sensitive with respect to its parameters. Adam, however, expresses stability and is among the first algorithms to converge to its best result (usually within its first 10 training epochs). Although RMSProp performs particularly well for station 42056,
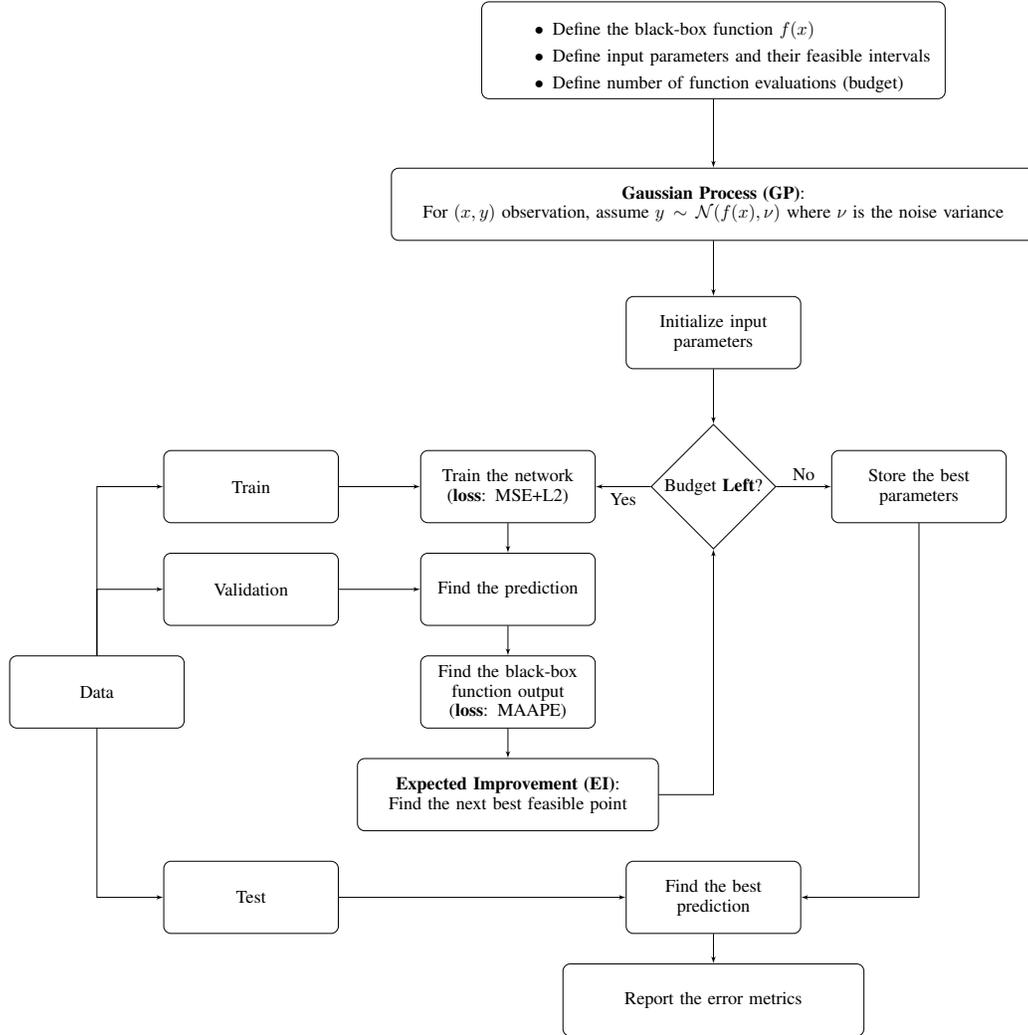
Figure 3: Hyperparameter tuning process with Spearmint Bayesian optimization (GP-EI-OptChooser) on black-box (seqtoseq) function

there are cases where the algorithm cannot reach the desired test error. Based on these results, from now on we use either Adam or AMSGrad as the optimizer.

4.2 Networks Comparison

In this section, we conduct an experiment to evaluate the performance of different neural network structures on the performance of the algorithm. We compare Single (Multi) layered RNN, Single (Multi) layered LSTM and epoch-scheduled seqtoseq model with Adam and AMSGrad optimization algorithms. RNN and LSTM networks contain a fully connected last layer. We consider seqtoseq model with both Adam and AMSGrad optimization algorithms.

Tables 3 and 4 demonstrate the performance of these neural network structures for 5- and 10-step-ahead forecasting of energy flux over data gathered from 5 refined buoys (collected from Table 1), using MAPE and MAAPE as the error metrics.

To evaluate the networks fairly, during hyperparameter tuning we used the same budget in Spearmint for each network. We set this budget equal to 100 function evaluations. One may argue that this experimental design might favor the alternate networks. Because they have many fewer hyperparameters to tune but for seqtoseq framework Spearmint deals with a more complex function to extract information from, having the same number of function evaluations.
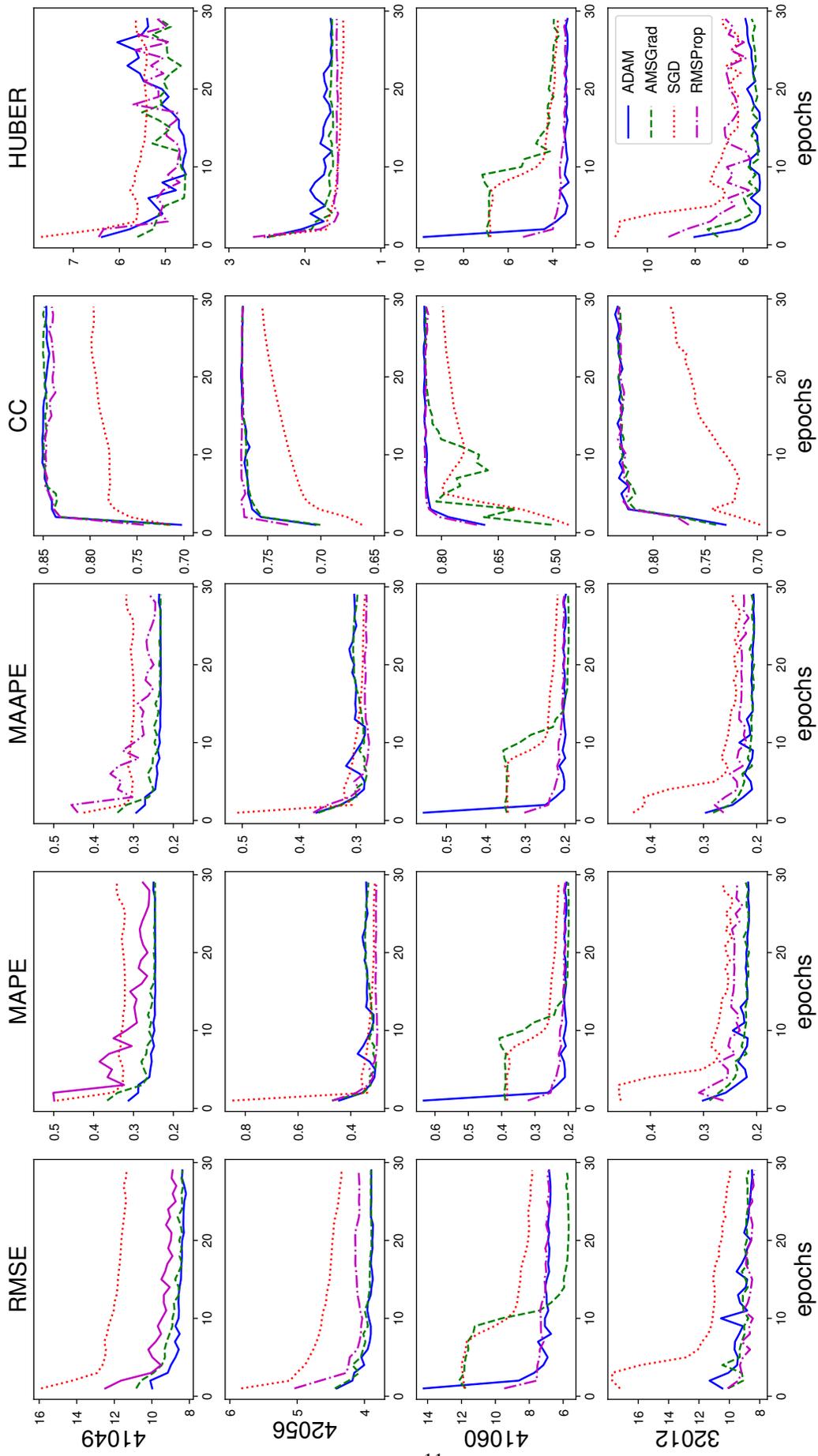
Figure 4: Test error comparison for four measurement stations. Learning rate: $\alpha = 0.001$ with decay=0.9 for all, Adam & AMSGrad: $\beta_1 = 0.9$ and $\beta_2 = 0.99$ and $\epsilon = 1e-8$, RMSProp: momentum=0.0, $\epsilon = 1e-10$.

11

Multi-layered networks consist of four layers. The first three layers are recurrent of structure [time steps, time steps, hidden size] with REctified Linear Unit (RELU) activation functions and the last layer is a fully connected one attaching the previous output to a layer of the last node. Training process has done with the budget of at most 15 epochs. In other words, if a structure reaches its best performance anywhere before 15th epoch, that would be collected. The best parameters for epoch-scheduled seqtoseq are $[64, 10, 16, 0.001, 1, 0.001, 0.4 \times \text{epochs}]$ in same order as shown in table 3.5.

From Table 3, one can see that seqtoseq networks result in smaller MAPE and MAAPE values, but single-layered LSTM plus a fully connected last layer also performs very well. We believe a fully connected last layer assists LSTM and simple RNN networks in capturing all learned features and translating them into superior forecasting. In addition, Multi layered structures are not superior to single layered ones. This happens in the Spearmint tuning process as well. Spearmint mostly prefers single or double layer structures for energy flux forecasting of any structures. Hence, when we force the stacked layers to be four, we can see a tiny reduction in the performance of the models. Moreover, Adam and AMSGrad perform closely to each other, and one cannot claim the superiority of any.

Table 3: 5-step-ahead energy flux forecasting, recurrent time step =10, hidden size=64, stacked recurrent layers: 1 or 4, batch size=16.

Station ID	SL-RNN+FCL		SL-LSTM+FCL		ML-RNN+FCL		ML-LSTM+FCL		Seqtoseq(Adam)		Seqtoseq (AMSGrad)	
	MAPE	MAAPE	MAPE	MAAPE	MAPE	MAAPE	MAPE	MAAPE	MAPE	MAAPE	MAPE	MAAPE
46013	0.296	0.259	0.295	0.258	0.303	0.266	0.301	0.265	**0.269**	**0.248**	0.297	0.271
41048	0.336	0.279	0.308	0.259	0.334	0.278	0.354	0.289	**0.217**	**0.205**	0.221	0.211
46084	0.395	0.306	0.392	0.304	0.397	0.307	0.399	0.308	0.311	0.279	**0.281**	**0.258**
46083	0.389	0.310	0.402	0.318	0.397	0.314	0.464	0.356	0.291	0.266	**0.271**	**0.253**
41060	0.215	0.193	0.221	0.197	0.209	0.189	0.201	0.180	0.174	0.169	**0.163**	**0.159**

In Table 4, we can observe that when we move from 5- to 10-step-ahead forecasting, epoch-scheduled seqtoseq fits the actual outputs even more accurately. For instance, at station 41048, seqtoseq provides a $(0.308 - 0.217)/0.217 = 41.9\%$ improvement over the best alternative method in terms of MAPE for 5-step-ahead forecasting, whereas for 10-step-ahead the improvement is $(0.459 - 0.299)/0.299 = 53.5\%$. Similarly, for MAAPE, the 5- and 10-step improvements are $(0.259 - 0.205)/0.205) = 26.3\%$ and $(0.333 - 0.278)/0.278 = 33.2\%$, respectively. These extra improvements may be a direct result of scheduled training managing deeper forecasting horizons.

Table 4: 10-step-ahead energy flux forecasting, recurrent time step =10, hidden size=64, stacked recurrent layers: 1 or 4, batch size=16.

Station ID	SL-RNN+FCL		SL-LSTM+FCL		ML-RNN+FCL		ML-LSTM+FCL		Seqtoseq(Adam)		Seqtoseq (AMSGrad)	
	MAPE	MAAPE	MAPE	MAAPE	MAPE	MAAPE	MAPE	MAAPE	MAPE	MAAPE	MAPE	MAAPE
46013	0.487	0.360	0.459	0.347	0.473	0.359	0.473	0.359	**0.376**	**0.328**	0.402	0.345
41048	0.470	0.336	0.459	0.333	0.707	0.470	0.691	0.450	0.436	0.345	**0.299**	**0.278**
46084	0.734	0.434	0.742	0.437	0.769	0.454	0.727	0.432	0.494	0.377	**0.393**	**0.341**
46083	0.592	0.399	0.569	0.393	0.624	0.410	0.602	0.406	0.425	0.360	**0.388**	**0.341**
41060	0.353	0.282	0.375	0.295	0.374	0.298	0.428	0.331	**0.207**	**0.199**	0.209	0.201

4.3 Long-Term Forecasting

Next, we investigate 2-day-ahead forecasting of H_s. As noted by [38], 2 days can be considered an acceptable range for long-term forecasting of H_s. Table 5 presents 48-step-ahead test error comparison over 5 refined buoys for H_s forecasting using the seqtoseq model.

Note the large gap between RMSE and HUBER losses that the model experiences when it predicts energy flux compared to significant wave height. The reason is the difference in the value range of output power and significant wave height. $\mathcal{P}$ is on the order of 10 kW/m, while H_s is usually on the order of 1 m. Hence, we focus on MAAPE. The results indicate that the model is robust for long-term prediction. The values presented in Table 5 are comparable to those for 5- and 10-step-ahead predictions for $\mathcal{P}$ in Section 4.2. Further, by definition, the HUBER metric should behave robustly with regard to outliers. The results support this fact and the model enjoys minimal HUBER loss values.

From the table, it is clear that Adam and AMSGrad produce very similar errors. Therefore, we conclude that one will not miss useful details by considering only the Adam optimization algorithm. Hence, in the last section, we optimize the models using Adam.

Table 5: 48 steps ahead H_s forecasting, recurrent time step =10, hidden size=64, stacked recurrent layers: 2, batch size=16.

Station ID	Seqtoseq (Adam)				Seqtoseq (AMSGrad)			
	RMSE (m)	MAPE	MAAPE	HUBER	RMSE (m)	MAPE	MAAPE	HUBER
41043	**0.593**	0.249	0.235	**0.147**	0.598	**0.247**	**0.233**	0.150
41040	0.431	**0.182**	0.177	**0.089**	0.431	0.183	**0.176**	0.090
41001	1.197	0.459	0.343	0.457	1.197	0.459	0.343	0.457
51002	0.513	0.186	0.180	0.126	**0.511**	0.186	**0.179**	**0.125**
46086	**0.658**	**0.238**	**0.226**	**0.177**	0.666	0.240	0.230	0.181

5 Reconstruction of Significant Wave Height

Our framework, on its surface, is not designed to reconstruct missing measurements of a station based on information from other stations; rather, it uses historical data at the same station. Therefore, to address this goal, we modify the input of the neural networks in such a way that the models will no longer require historical data. Figure 5 illustrates three buoys—46069, 46025 and 46042—that are close to one another. For more details on buoys you can see the Table 1. Similar to existing methods [2, 3], we aim to reconstruct the significant wave height of buoy 46069 at different time steps, treating them as missing data. Networks have access to all information from the two adjacent buoys, 46025 and 46042, as their input, as well as buoy 46069's SWH as their labels in the training process, which consists of data from the entire year 2009. Then, the networks predict the entire 2010 year SWH of buoy 46069, based on the inputs from the adjacent buoys. In other words, at time step t, each model is allowed to see the first t measurements of H_s at the two nearby buoys.

Table 6 compares the performance of our seqtoseq and SL LSTM networks with that of benchmark methods from the literature. The methods are [2, 3]: 1) All-featured Extreme Learning Machine, (ELM) 2) All-featured Support Vector Regression (SVR), 3) Grouping Genetic Algorithm Extreme Learning Machine with final prediction with ELM (GGA-ELM-ELM), 4) Grouping Genetic Algorithm Extreme Learning Machine with final prediction with SVR (GGA-ELM-SVR), 5) Bayesian optimization Grouping Genetic Algorithm Extreme Learning Machine with final prediction of ELM (BO-GGA-ELM-ELM) and 6) Bayesian optimization Grouping Genetic Algorithm Extreme Learning Machine with final prediction of SVR (BO-GGA-ELM-SVR). The table reports the RMSE, mean absolute error (MAE), and r^2 (which is CC^2) as error metrics; these are the metrics used in the benchmark papers.

Note that [2, 3] uses buoys 46025, 46042, and 46069, and claims that the NOAA dataset has no missing measurements for these buoys in the years 2009–10. However, we found this to be untrue. First, there are significant gaps in the 46069 station data, which result in fewer than 5000 data points instead of the 8760 hourly points that should be present in a given year. Second, like the other refined stations, roughly 1–2% of the H_s values contain 99.00, which indicates missing data. Therefore, we carefully cleaned the missing data points from all three stations in such a way as to preserve the position of each time step. That is, if only there is a missing point in a station measurement, we exclude that time step from all stations. The resulting data set has 9263 data points for each buoy, where 4687 points are for training (2009) and 4576 are for testing (2010).

Table 6: Comparison of different methods for reconstructing H_s of station 46069, epochs ≤ 50.

Methods	RMSE (m)	MAE (m)	r^2
All features-ELM	0.4653	0.3582	0.6624
All features-SVR	0.6519	0.4986	0.3949
GGA-ELM-ELM	0.3650	0.2858	0.7049
GGA-ELM-SVR	0.3599	0.2727	0.7056
BO-GGA-ELM-ELM	0.3324	0.2519	0.7429
BO-GGA-ELM-SVR	0.3331	0.2461	0.7396
Seqtoseq (Adam)	0.3419	0.2528	**0.8106**
LSTM+FCL (Adam)	**0.2784**	**0.2392**	0.7922

From the table, it is clear that the performance of the seqtoseq network is very promising, as it obtains the best r^2 values among the methods. Moreover, LSTM+FCL significantly outperforms all the alternatives and even the seqtoseq network in terms of RMSE and MAE simultaneously. It is worth mentioning that both of our methods have access only to H_s of the stations, while the benchmark methods use more measurements. For example, the GAA methods obtain their best performance using wind speed, significant wave height, average wave period, air temperature and the atmospheric pressure at buoy 46025 and the significant wave height and average wave period of buoy 46042 [2]. This
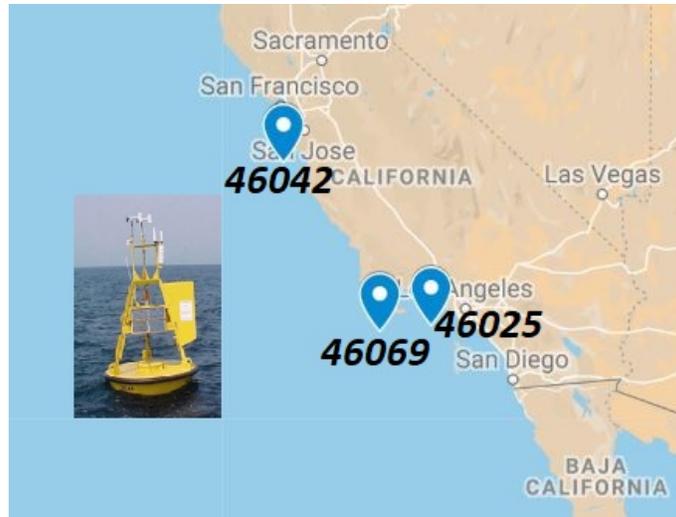
Figure 5: Three adjacent buoys of 46069, 46042 and 46025

makes the superiority of our methods even clearer, but brings about another question, as well, namely: Would using proper feature selection improve the performance of deep networks even further? We conducted some preliminary experiments to answer this question, considering combinations of the features that the GAA approaches use, and found an inferior accuracy compared to solely using H_s. In Section 6, we discuss a more efficient approach for feature selection.
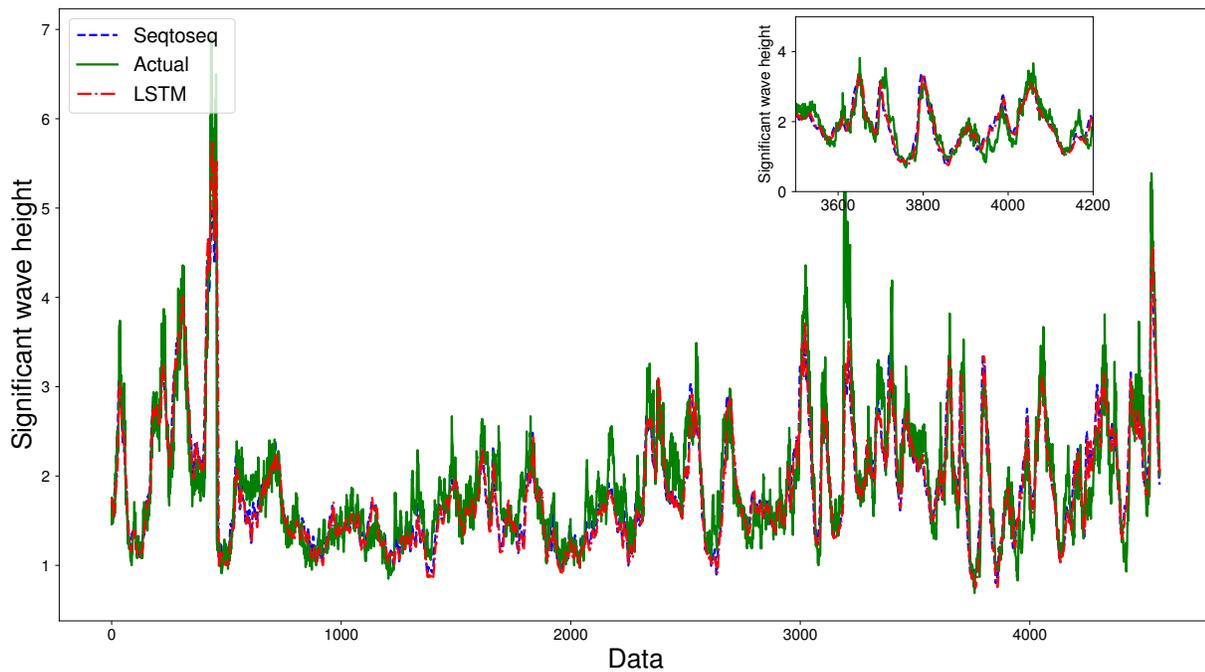
Figure 6: Significant wave height reconstruction of buoy 46069. Inset shows a zoomed-in portion.

14

Figure 6 illustrates the performance of both the LSTM and seqtoseq networks in reconstructing the significant wave height for buoy 46069. Note that the models never "saw" the solid green line, which is the year 2010 data (i.e., the test data).

We also illustrate the RMSE behavior of LSTM+FCL and seqtoseq. Figure 7 displays the RMSE test errors based on the number of training epochs up to 50. Both networks reach their best performance around tenth epoch with sequence-to-sequence reaching slightly faster. This happens even though there are more weights in the sequence-to-sequence model. We argue that epoch-scheduling training may be responsible for this quicker training performance, because by definition it allows the network to see ground truth outputs to update the parameters better while simultaneously avoiding excessive overfittings.
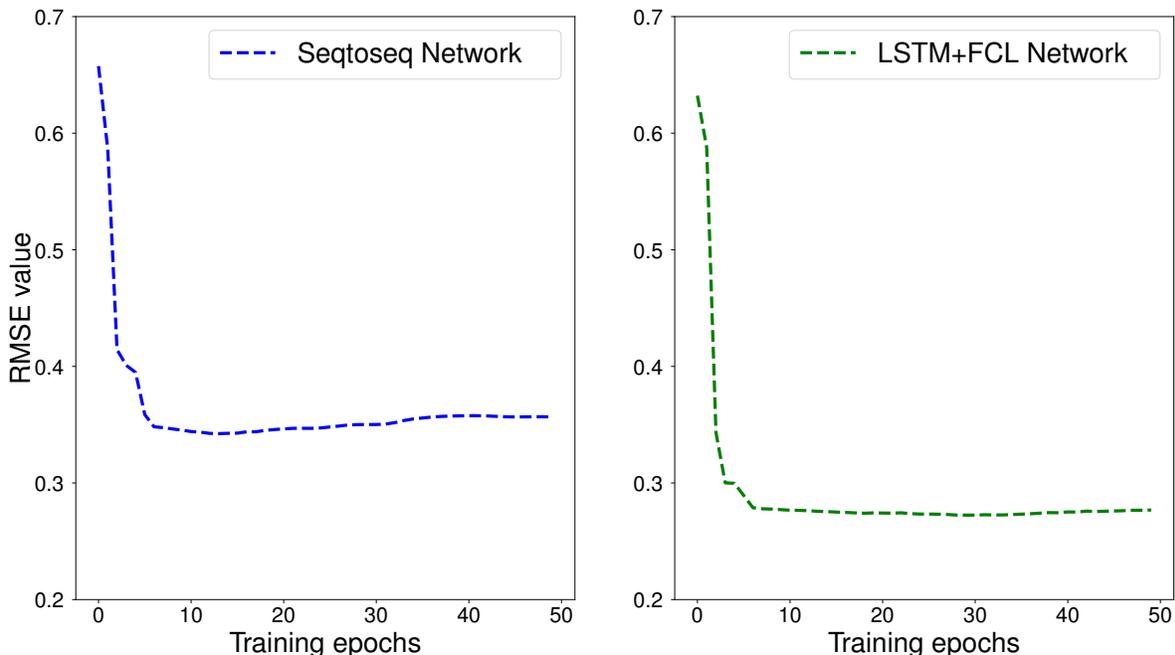

Figure 7: H_s RMSE test (year 2010) behavior on training data (year 2009) of buoy 46069.

6 Feature Selection

In this section, we modify the proposed framework to investigate feature selection via elastic net [57] methodology. We alter the loss function to include two separate regularizers, "ℓ_1 norm" and "ℓ_2 norm", as follows:

$$f_w(x) = \frac{1}{n} \sum_{i=1}^{n} \left(y^i - N(x^i; w) \right)^2 + \lambda_1 \|w\|_2^2 + \lambda_2 \|w\|_1$$

where $\|w\|_1 = \sum_W w_i$ and W is the set of all the network weights. The "ℓ_1 norm" inherits a naturally sparse collection of variables. $\|w\|_1$ has its minimum when all the weights are zero (uncollected) so every single nonzero weight should improve the first term of the objective function. Hence, the model chooses only those weights for which the decrease in the first term outweighs the increase in the last term. Using only the "ℓ_1 norm" regularizer would result in instability among the multiple solutions because a tiny change in the parameters may change some weights from zero to nonzero values or vice versa [58]. Therefore, the "ℓ_2 norm" is used in the objective.

One important difference between our feature selection method and those used in the wave energy literature (e.g., [3]) is that the models in the literature either use all of the data related to a specific feature, like H_s, from a nearby buoy, or decide to ignore that feature entirely; however, our model allows us to partially utilize any of the features based on nonzero weights resulting in the optimal solution.

15

Table 7: Comparison of different methods reconstructing H_s and $\mathcal{P}$ of 44008 station, epochs ≤ 50.

| | Wave Feature | | | | | | | |
| | H_s | | | | $\mathcal{P}$ | | | |
Methods	RMSE (m)	HUBER	MAAPE	Non-Zero Var(%)	RMSE (m)	HUBER	MAAPE	Non-Zero Var(%)
SL-LSTM+FCL	0.569	0.209	0.301	99.4	3.031	8.394	0.520	99.2
ML-LSTM+FCL	0.391	0.120	0.304	99.7	3.065	9.041	0.510	99.9
Seqtoseq	0.469	0.108	0.279	99.7	4.593	2.229	0.486	99.7
SL-LSTM+FCL +EN	0.439	0.135	0.354	20.0	3.171	7.543	0.500	16.2
ML-LSTM+FCL +EN	0.368	0.110	0.289	33.6	**2.993**	9.224	0.500	50.7
Seqtoseq + EN	0.459	0.106	**0.264**	58.2	4.526	**2.211**	**0.455**	69.8
Random Forrest	**0.359**	**0.055**	0.269		3.431	5.618	0.561	

To investigate the performance of our proposed framework for feature collection, we design a new experiment which has been recently explored for feature selection via ensemble structures as well [59]. Figure 8 illustrates six stations—44007, 44008, 44009, 44013, 44014 and 44017—from the east coast of the United States. To see their exact locations and water depths, you can refer to Table 1. We reconstruct the significant wave height and power output of buoy 44008 based on available features of the other nearby buoys. We chose these buoys for two main reasons. First, on average they are rich in features with usable data. Second, the resolution and the data-collection times for these stations are the same up to 1-minute accuracy. This is crucial when we want to reconstruct a feature of a buoy based on others. We consider wind direction (WDIR), wind speed (WSPD), gust speed (GST), significant wave height (WVHT), dominant wave period (DPD), average wave period (APD), direction of dominant period waves (MWD), sea level pressure (PRES), air temperature (ATMP), sea surface temperature (WTMP) and dewpoint temperature (DEWP). The datasets are for the year 2018. We include all the features for all the buoys, except DEWP for buoys 44009 and 44013 because those data are unusable. We exclude a measurement for all the stations only if a feature is missing in at least one of them. In the end, the dataset consists of 3422 measurements of 53 features. We divide the dataset into train (60%), validation (20%) and test (20%). We do not change the real measurements in any way. In addition, the methods never "see" the test set during training and validation. In this experiment, for both the LSTM+FCL and seqtoseq networks we have one more parameter to tune, namely, the coefficient of $\mathbb{L}1$-norm) during the validation.

Table 7 compares the performance of all of the methods, along with an alternative method, Random Forest, for reconstructing both H_s and $\mathcal{P}$ wave features. There is a column dedicated to the percentage of non-zero trainable variables. We consider a single weight or bias variable to be non-zero if its absolute value exceeds the threshold of 0.0001, and we report the result as the percentage of non-zero variables among all trainable variables. When we use the elastic net concept for selecting weights and biases, we observe a significant drop in the non-zero values. This means exactly what we discussed at the beginning of this section. The models invest only in those variables which are beneficial to the objective value. Here, we report the best results found for each measurement error RMSE, HUBER and MAAPE independently. In other words, the reported values of RMSE, HUBER or MAAPE are not necessarily gathered from a single experiment, but instead they are the best ones for the training epochs less than or equal to 50. As seen in Table 7, EN generally improves the performance of each method. For instance, EN aids single-layered LSTM+FCL significantly for reconstructing H_s and reduces RMSE and HUBER losses by 29% and 54%, respectively. Similarly, it helps the seqtoseq network slightly and reduces the measurement errors by around 5%.

Another interesting finding is the comparison between multi-layered LSTM+FCL and single-layered ones. Unlike in the prediction section 4.2, here the performance of multi-layered LSTM+FCL is satisfactory, and even superior in many cases. Therefore, we argue that for feature selection, when the number of features is considerable compared to forecasting, having deeper structures improves the performance.

In addition, we observe that networks with more variables, such as multi-layered LSTM+FCL or seqtoseq, retain more non-zeros. In particular, in the process of tuning, they choose the $\mathbb{L}1$-norm coefficient to be smaller. Hence, the effect of the $\mathbb{L}1$-norm is lessened. This may be due to the fact that the more advanced structures can find hidden relations among features and set their corresponding weights to non-zero values, whereas single-layered models are unable to discover some hidden relations, so they set the weights equal to zero.

The Random Forest algorithm [60] is known to be very efficient and accurate in ranking the importance of variables. To have a fair comparison, we hypertune its major parameters such as maximum depth. For H_s reconstruction, it finds the best RMSE and HUBER losses, but for $\mathcal{P}$ reconstruction, it fails to provide similar performance. The seqtoseq+EN framework is the most robust and promising one. It attains the best performance for half the measurements and is generally among the top methods for the rest. We observe that $\lambda_i \in [0.001, 0.01]$, $i = 1, 2$ results in the leading performance for the seqtoseq network.
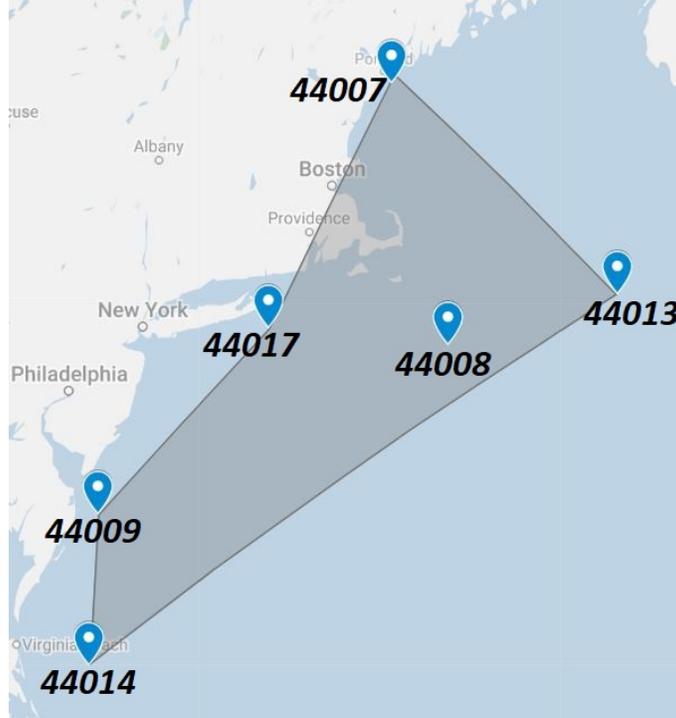
Figure 8: Adjacent buoys of 44007, 44008, 44009, 44013, 44014 and 44017

7 Conclusion

This article focuses on the reconstruction, feature selection and multivariate, multistep forecasting of ocean wave characteristics based on real data obtained from NOAA buoy measurements around the globe. This paper is among the first to propose the use of sequence-to-sequence and other novel recurrent networks for these objectives. Moreover, the epoch-scheduled training concept has been introduced as a soft technique so that the model enjoys the consistency of teacher forcing methods while avoiding overfitting. We tested various optimization algorithms on the networks that we introduced. AMSGrad and Adam present robust and promising performance comparing with SGD and RMSProp. We compared several recurrent networks. All of the parameters are tuned with Spearmint Bayesian optimization under the same budget. One observes that Spearmint favors single-layered networks as opposed to multi-layered networks for multivariate forecasting. This can be observed in our numerical studies as well. The SL-LSTM+FCL and seqtoseq models demonstrate superior performance compared to the other networks.

Furthermore, we explore the problem of reconstructing wave features, which has been well studied in the literature. The results suggest the superiority of our proposed networks compared to existing techniques. Our findings emphasize the fact that using only significant wave heights of adjacent buoys is sufficient to construct H_s of a nearby station. Proper feature selection, however, necessitates more analysis. We design a new experiment using NOAA data from the east coast of the United States. The dataset can be used for any feature selection and multivariate regression research. We incorporated the elastic net concept into our proposed neural networks to handle 53 features of the dataset. We evaluate the performance with and without the EN and with the random forest algorithm. The results indicate that the seqtoseq network has a consistent and reliable performance. In addition, for feature selection, deeper recurrent structures are more promising compared to single-layered ones.

We suggest the use of different parameter tuning algorithms such as derivative-free optimization instead of Spearmint as one promising future direction. Moreover, utilizing other neural network techniques—such as attention mechanism, in particular—for wave feature selection and reconstruction can be another area for future study.

References

[1] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[2] L Cornejo-Bueno, JC 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.

[3] Laura Cornejo-Bueno, Eduardo C Garrido-Merchán, Daniel Hernández-Lobato, and Sancho Salcedo-Sanz. Bayesian optimization of a hybrid system for robust ocean wave features prediction. *Neurocomputing*, 275: 818–828, 2018.

[4] A Hadjihosseini, J Peinke, and NP Hoffmann. Stochastic analysis of ocean wave states with and without rogue waves. *New Journal of Physics*, 16(5):053037, 2014.

[5] Igor Rychlik, Pär Johannesson, and Malcolm R Leadbetter. Modelling and statistical analysis of ocean-wave data using transformed gaussian processes. *Marine Structures*, 10(1):13–47, 1997.

[6] Malcolm John Tucker and Edward G Pitt. *Waves in ocean engineering*. Number Volume 5. 2001.

[7] Johannes Falnes. A review of wave-energy extraction. *Marine structures*, 20(4):185–201, 2007.

[8] Kenneth Edward Steele and Theodore Mettlach. Ndbc wave data—current and planned. In *Ocean Wave Measurement and Analysis*, pages 198–207. ASCE, 1993.

[9] Hendrik L Tolman. A third-generation model for wind waves on slowly varying, unsteady, and inhomogeneous depths and currents. *Journal of Physical Oceanography*, 21(6):782–797, 1991.

[10] The Wamdi Group. The wam model—a third generation ocean wave prediction model. *Journal of Physical Oceanography*, 18(12):1775–1810, 1988.

[11] Klaus Hasselmann, TP Barnett, E Bouws, H Carlson, DE Cartwright, K Enke, JA Ewing, H Gienapp, DE Hasselmann, P Kruseman, et al. Measurements of wind-wave growth and swell decay during the joint north sea wave project (jonswap). *Ergänzungsheft 8-12*, 1973.

[12] W Perrie, H Günther, W Rosenthal, and B Toulany. Modelling wind-generated surface gravity waves using similarity in a coupled discrete wave model. *Quarterly Journal of the Royal Meteorological Society*, 115(490): 1373–1396, 1989.

[13] NRRC Booij, RC Ris, and Leo H Holthuijsen. A third-generation wave model for coastal regions: 1. model description and validation. *Journal of geophysical research: Oceans*, 104(C4):7649–7666, 1999.

[14] Mehmet Özger and Zekai Şen. Prediction of wave parameters by using fuzzy logic approach. *Ocean Engineering*, 34(3-4):460–469, 2007.

[15] MH Kazeminezhad, A Etemad-Shahidi, and SJ Mousavi. Application of fuzzy inference system in the prediction of wave parameters. *Ocean Engineering*, 32(14-15):1709–1725, 2005.

[16] Roslan Hashim, Chandrabhushan Roy, Shervin Motamedi, Shahaboddin Shamshirband, and Dalibor Petković. Selection of climatic parameters affecting wave height prediction using an enhanced takagi-sugeno-based fuzzy methodology. *Renewable and Sustainable Energy Reviews*, 60:246–257, 2016.

[17] J Mahjoobi and Ehsan Adeli Mosabbeb. Prediction of significant wave height using regressive support vector machines. *Ocean Engineering*, 36(5):339–347, 2009.

[18] Mc C Deo, A Jha, AS Chaphekar, and K Ravikant. Neural networks for wave forecasting. *Ocean engineering*, 28(7):889–898, 2001.

[19] P Abhigna, S Jerritta, R Srinivasan, and V Rajendran. Analysis of feed forward and recurrent neural networks in predicting the significant wave height at the moored buoys in bay of bengal. In *Communication and Signal Processing (ICCSP), 2017 International Conference on*, pages 1856–1860. IEEE, 2017.

[20] R Savitha, Abdullah Al Mamun, et al. Regional ocean wave height prediction using sequential learning neural networks. *Ocean Engineering*, 129:605–612, 2017.

[21] Iman Malekmohamadi, Mohammad Reza Bazargan-Lari, Reza Kerachian, Mohammad Reza Nikoo, and Mahsa Fallahnia. Evaluating the efficacy of svms, bns, anns and anfis in wave height prediction. *Ocean Engineering*, 38(2-3):487–497, 2011.

[22] Robert Azencott, Viktoria Muravina, Rasoul Hekmati, Wei Zhang, and Michael Paldino. Automatic clustering in large sets of time series. In *Contributions to Partial Differential Equations and Applications*, pages 65–75. Springer, 2019.

[23] Hossein Kamalzadeh, Abbas Ahmadi, and Saeid Mansour. A shape-based adaptive segmentation of time-series using particle swarm optimization. *Information Systems*, 67:1–18, 2017.

[24] Andreas Uihlein and Davide Magagna. Wave and tidal current energy–a review of the current state of research beyond technology. *Renewable and Sustainable Energy Reviews*, 58:1070–1081, 2016.

[25] Chong Wei Zheng, Chong Yin Li, Jing Pan, Ming Yang Liu, and Lin Lin Xia. An overview of global ocean wind energy resource evaluations. *Renewable and Sustainable Energy Reviews*, 53:1240–1251, 2016.

[26] Chong Wei Zheng, Qing Wang, and Chong Yin Li. An overview of medium-to long-term predictions of global wave energy resources. *Renewable and Sustainable Energy Reviews*, 79:1492–1502, 2017.

[27] Sunil Kr Jha, Jasmin Bilalovic, Anju Jha, Nilesh Patel, and Han Zhang. Renewable energy: Present research and future scope of artificial intelligence. *Renewable and Sustainable Energy Reviews*, 77:297–317, 2017.

[28] Pooja Jain and MC Deo. Neural networks in ocean engineering. *Ships and offshore structures*, 1(1):25–35, 2006.

[29] Markel Penalba, Giussepe Giorgi, and John V Ringwood. Mathematical modelling of wave energy converters: a review of nonlinear approaches. *Renewable and Sustainable Energy Reviews*, 78:1188–1207, 2017.

[30] Marcus Lehmann, Farid Karimpour, Clifford A Goudey, Paul T Jacobson, and Mohammad-Reza Alam. Ocean wave energy in the united states: Current status and future perspectives. *Renewable and Sustainable Energy Reviews*, 2017.

[31] Kostas Hatalis, Parth Pradhan, Shalinee Kishore, Rick S Blum, and Alberto J Lamadrid. Multi-step forecasting of wave power using a nonlinear recurrent neural network. In *PES General Meeting— Conference & Exposition, 2014 IEEE*, pages 1–5. IEEE, 2014.

[32] Kumars Mahmoodi, Hassan Ghassemi, and Hashem Nowruzi. Data mining models to predict ocean wave energy flux in the absence of wave records. *49 Scientific Journals of the Maritime University of Szczecin*, (49):119–129, 2017.

[33] N Krishna Kumar, R Savitha, and Abdullah Al Mamun. Ocean wave height prediction using ensemble of extreme learning machine. *Neurocomputing*, 2017.

[34] Scott C James, Yushan Zhang, and Fearghal O'Donncha. A machine learning framework to forecast wave conditions. *arXiv preprint arXiv:1709.08725*, 2017.

[35] SWAN Team. Swan user manual. swan cycle iii version 40.91. *Delft University of Technology Technical documentation*, page 123, 2014.

[36] SN Londhe, Shalaka Shah, PR Dixit, TM Balakrishnan Nair, P Sirisha, and Rohit Jain. A coupled numerical and artificial neural network model for improving location specific wave forecast. *Applied Ocean Research*, 59: 483–491, 2016.

[37] WY Duan, Y Han, LM Huang, BB Zhao, and MH Wang. A hybrid emd-svr model for the short-term prediction of significant wave height. *Ocean Engineering*, 124:54–73, 2016.

[38] Pradnya Dixit and Shreenivas Londhe. Prediction of extreme wave heights using neuro wavelet technique. *Applied Ocean Research*, 58:241–252, 2016.

[39] R Prahlada and Paresh Chandra Deka. Forecasting of time series significant wave height using wavelet decomposed neural network. *Aquatic Procedia*, 4:540–547, 2015.

[40] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[41] Zachary C Lipton, John Berkowitz, and Charles Elkan. A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019*, 2015.

[42] Aryan Mobiny and Mohammad Najarian. Text-independent speaker verification using long short-term memory networks. *arXiv preprint arXiv:1805.00604*, 2018.

[43] Chung-Cheng Chiu, Tara N Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjuli Kannan, Ron J Weiss, Kanishka Rao, Katya Gonina, et al. State-of-the-art speech recognition with sequence-to-sequence models. *arXiv preprint arXiv:1712.01769*, 2017.

[44] Ronald J Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.

[45] Alex M Lamb, Anirudh Goyal ALIAS PARTH GOYAL, Ying Zhang, Saizheng Zhang, Aaron C Courville, and Yoshua Bengio. Professor forcing: A new algorithm for training recurrent networks. In *Advances In Neural Information Processing Systems*, pages 4601–4609, 2016.

[46] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 1171–1179, 2015.

[47] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. 1999.

[48] Feiping Nie, Heng Huang, Xiao Cai, and Chris H Ding. Efficient and robust feature selection via joint l2, 1-norms minimization. In *Advances in neural information processing systems*, pages 1813–1821, 2010.

[49] Jun Liu, Shuiwang Ji, and Jieping Ye. Multi-task feature learning via efficient l 2, 1-norm minimization. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*, pages 339–348. AUAI Press, 2009.

[50] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[51] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. 2018.

[52] L Cuadra, S Salcedo-Sanz, JC Nieto-Borge, E Alexandre, and G Rodríguez. Computational intelligence in wave energy: Comprehensive review and case study. *Renewable and Sustainable Energy Reviews*, 58:1223–1246, 2016.

[53] N Krishna Kumar, R Savitha, and Abdullah Al Mamun. Ocean wave height prediction using ensemble of extreme learning machine. *Neurocomputing*, 277:12–20, 2018.

[54] N Krishna Kumar, R Savitha, and Abdullah Al Mamun. Ocean wave characteristics prediction and its load estimation on marine structures: A transfer learning approach. *Marine Structures*, 61:202–219, 2018.

[55] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959, 2012.

[56] Kevin Swersky, Jasper Snoek, and Ryan P Adams. Multi-task bayesian optimization. In *Advances in neural information processing systems*, pages 2004–2012, 2013.

[57] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, 67(2):301–320, 2005.

[58] Andrew Y Ng. Feature selection, l 1 vs. l 2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, page 78. ACM, 2004.

[59] Mohammad Pirhooshyaran, Katya Scheinberg, and Lawrence V Snyder. Feature engineering and forecasting via derivative-free optimization and ensemble of sequence-to-sequence networks with applications in renewable energy. *Energy*, page 117136, 2020.

[60] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.