IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 12, 2021, accepted January 31, 2021, date of publication February 10, 2021, date of current version February 19, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3058592

Using Neural Network Approaches to Detect Mooring Line Failure

AMIR MUHAMMED SAAD[1,2], FLORIAN SCHOPP[1,2], RODRIGO A. BARREIRA[3], ISMAEL H. F. SANTOS[3], EDUARDO A. TANNURI[2], EDSON S. GOMI[2], (Senior Member, IEEE), AND ANNA H. REALI COSTA[1,2], (Member, IEEE)

[1]Intelligent Techniques Laboratory, Universidade de São Paulo (USP), Sao Paulo 05508-900, Brazil
[2]Numerical Offshore Tank, Universidade de São Paulo (USP), Sao Paulo 05508-900, Brazil
[3]Petrobras, Rio de Janeiro 20031-912, Brazil

Corresponding author: Amir Muhammed Saad (amir.saad@usp.br)

This work was supported in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), Brazil, under Finance Code 001; in part by the Petroleum Development Technology Fund of Nigeria, Nigeria; in part by the ANP/PETROBRAS, Brazil, under Project N. 21721-6; and in part by the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) under Grant 425860/2016-7, Grant 307027/2017-1, and Grant 304784/2017-6.

ABSTRACT The mooring systems give stability to the floating platforms against environmental conditions, stabilizing the platform with mooring lines attached to the seabed. The mooring systems are among the main components that guarantee the safety of the staff and the various operations carried out on the platforms. The current approaches used to monitor mooring lines are inefficient as line tension sensors are expensive to install, maintain, and have durability problems. This article presents the development of two neural network-based machine learning systems: a Multilayer Perceptron (MLP) and a Long Short-Term Memory (LSTM). They are able to detect mooring line failure in near real-time based on the comparison between measured and predicted motion. The implemented systems were trained and evaluated with simulated motion data generated using real environmental conditions measured in the Campos Basin, in Rio de Janeiro, Brazil. The results showed the MLP and LSTM models were able to detect a failure in the mooring lines, with increasing difference between the predicted and the measured motions when there is a line breakage. A comparison between the two machine learning models revealed the LSTM model performed better at predicting the motions of the platform.

INDEX TERMS Mooring line failure, failure detection, machine learning, neural networks, floating production storage and offloading.

I. INTRODUCTION

The discovery of large offshore oil fields has led the oil sector to find a solution to the challenge of stationkeeping in deepwater. Floating structures, also known as platforms such as Floating Production Storage and Offloading (FPSO), with large tank capacity, have been used over the years to extract offshore oil deposits safely. These floating platforms allow the extraction and storage of the extracted crude oil and are maintained in the desired location using mooring lines anchored to the seafloor. Mooring systems are among the key components that ensure the position keeping, the staff's safety, and the various operations carried out on a platform. These operations include extraction, production, and oil offloading. The need for reliable monitoring of the structure and integrity of the platform's anchoring system becomes critical because system failure can lead to the platform drifting from the desired location, causing endangerment to personnel aboard or around the platform, oil spillage, and environmental pollution. The floating structure exposed to offshore environmental conditions, such as waves, currents and wind, continuously undergoes stresses and strains caused by these environmental conditions. In this way, the platforms' structural integrity and mooring lines are decreased during the life cycle of these components.

Studies in [12] and [6] have shown around 45% of mooring failures are either single line or multiple line failures that can be attributed to corrosion and fatigue, among other causes. In some cases of single line mooring failure, additional mooring lines can be damaged due to increased load, stress, and tension experienced by the remaining lines, since single line

The associate editor coordinating the review of this manuscript and approving it for publication was Byung Cheol Song.

failure increases the degradation rate of the remaining mooring lines [12].

A report in [1] indicated that 50% of the offshore platforms in the North Sea could not monitor the mooring line in real-time, 33% of the platforms cannot measure offset from the no-load equilibrium, and 78% of the platforms lack a system that alerts in the event of a line failure. Therefore, when a mooring system gets comprised, it can go unnoticed for an extended period. Identifying mooring line damage is predominantly done via visual inspection of the mooring line in a scheduled time frame. However, this approach is rudimentary, time-consuming, and costly. To overcome this problem, methods that involve monitoring of the mooring line have been proposed. In these methods, inclinometers, micro-remote operated vehicles (ROV), load shackles are used. Inclinometers are fitted on the mooring lines to measure the mooring line angles. In calm weather conditions, mooring angles are measured and compared against mooring angles after turbulent weather has passed. In the event of significant changes in the angles between the two measurements taken, this may indicate the possibility of a failure in a mooring line [19]. Micro-ROVs are also deployed to check the mooring line if the inclinometer reading shows an offset. The use of micro-ROVs to inspect anchor lines helps to eliminate the need to employ full-size, expensive ROVs or even offshore divers when an event occurs. However, as these platforms venture into greater water depth the mooring line length increases. Moreover the image quality of the micro-ROV increasingly becomes blurred [14].

Load shackle with load cells are connectors used to link mooring lines. These load shackles monitor mooring line tension in real-time with the use of load cell [8]. There are different types of load shackles, e.g., wireless and traditional load shackles. Wireless load shackles require interval replacement of the battery that serves as the energy source in deep waters. Both traditional and wireless shackles are affected by marine growth that occurs naturally on the mooring line which affects the tension load on the monitored mooring line. Removal of marine growth is expensive and time-consuming. It is also costly to place these sensors along the length of the mooring line in deep water.

Position monitoring can be achieved using an orbital satellite, such as a Differential Global Positioning System (DGPS), to monitor a platform's position. The platform is monitored continuously, as the navigation system onboard the platform coupled to a computer is in constant communication with the DGPS sensor. Platforms that drift away from the perimeter of a predefined watch circle most likely have encountered mooring line failure [19]. However, the watch circle's definition is not always easy to do, as it must take into account the different geographical locations under specified environmental conditions, which can cause difficulties for this approach to provide satisfactory responses in the most extreme environmental conditions.

Recent improvements in technology feature solutions based on machine learning (ML) for real-time monitoring and

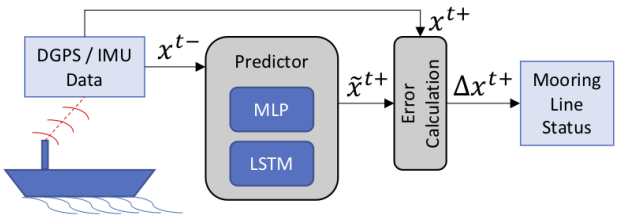


FIGURE 1. Proposed general architecture: mooring line failure detection is based on the short-term prediction of platform motion.

failure detection of the mooring system. ML, a subset of Artificial Intelligence in which the models are built based on data by an algorithm with no explicit instructions being provided, becomes a good alternative for use in monitoring mooring systems. The abundance of data generated by several sensors can be used to train the ML algorithms in this task. Platform motions in the offshore with an intact mooring system and a single mooring line damage show subtle variation in their response to environmental conditions making it challenging to know when mooring line failure has occurred assertively. ML models that are useful for identifying subtle differences in intricate patterns were implemented to detect these changes.

Our proposal in this article is to use architecture as shown in Fig. 1, with a predictor module that estimates the future platform motion based on the platform's previous motion data. This predictor is trained with data that indicates motions without line breakage. The idea is that the predictor makes the future prediction of the motion considering the absence of failures and that the sensors measure the actual motion performed by the platform. If there is a significant difference between the predicted and the measured value, this is notified by the mooring line's status, which then indicates a line failure. We present the development and comparison of two ML models to identify when a platform's mooring system is compromised, based on neural networks (NN) for the predictor module, a Multi-layer Perceptron (MLP) model, and a Long Short Term Memory (LSTM) model.

This article is structured as follows: Section II introduces related work. Section III introduces key concepts of machine learning. Section IV explains in detail our proposal, while Section V describes the experimental setup and describes the experiments and related analysis. Section VI presents result of the experiments carried out and discussion. Finally, Section VII presents our conclusions and highlights future work.

II. RELATED WORK

Researchers in diverse domains are increasingly adopting machine learning techniques, which is also true for the offshore industry. The articles included in this section focus mainly on the detection of mooring line failures for FPSO, given that this platform is the interest of our study. Table 1 presents eight articles published in the last five years retrieved with query "(machine AND learning OR neural AND

network OR artificial)'' to the Scopus database. The selected papers use neural networks as their primary ML technique, and the table presents a summary of the neural networks the authors developed and the inputs fed to their respective models.

In the following, we discuss how these ML techniques were applied to monitoring the mooring systems. We detail the input variables used, the algorithm implemented, and the output of each algorithm.

MLP and kriging methods was used in [9] to predict changes in the mooring state. Both MLP and kriging methods were trained on meta-ocean data and GPS data. Inputs to their models included wave height, period and direction, wind velocity, current velocity, draft of the vessel, and outputs of the models were the mean offset (from the mooring design center), maximum offset, significant offset and second-order peak response period (Natural Frequency). The dataset used in training methods had instances in which mooring failure had occurred and instances of intact mooring lines. The models were trained on a dataset without line breakage and tested on the dataset with line breakage. They then used the error difference between the dataset without line breakage and line breakage to determine the occurrence of failure. The result showed both methods performed well in predicting when changes in the mooring state occur.

Also, in [15] a novel concept with regards to the integrity of the mooring system was proposed. They implemented a system named Position Response Learning System (PRLS) that, at its core, uses MLP for predicting the integrity of a mooring system. The PRLS system was trained on data from DGPS, meta-ocean sensors (waves, currents, and wind), and inertial motion of the vessel; they used the six degrees of freedom (6DoF) motion sensors of the vessel, namely surge, sway, heave, roll, pitch and yaw as input features. The proposed system provided two forms of output: classification-based output, indicating whether or not there was a line break, and regression-based output, estimating which line could have broken.

A two hidden layer backpropagated MLP on numerically simulated data to predict mooring line tension from platform motion was also used in [16]. MLTSIM, a proprietary numerical simulation package, was used to model a semi-submersible vessel with its mooring system. Properties of the vessel and its mooring line material composition were accounted for in the numerical simulation under different sea state conditions. Results from the simulation were used as data for training and testing of the MLP. 60-second time series of platform movements with one-second intervals were used as an input feature for MLP. Series of 30 seconds of mooring line tension with one-second intervals were predicted as output. The MLP training performance was evaluated using the average RMSE error. Evaluation of the MLP was carried out using the correlation coefficient to check for the accuracy of the MLP model prediction. Comparison between the predicted tension of the mooring line of the MLP model and the output value of the numerical simulation of the mooring line tension was performed during the training and test phases using the correlation coefficient. Results in both the training and testing phases of the MLP showed high correlation values indicating in the training phase that the model was able to learn, while in the testing phase, the model was able to generalize on what it learned to new unseen data. In [17], an MLP model capable of detecting when a moored offshore platform mooring line experiences damage to its mooring lines was implemented. The presented MLP was a two hidden layer model trained to distinguish between normal drift period of a moored platform readings from damaged drift readings of a moored platform. The proposed MLP was trained on the results of an in-house numerical simulation software named MLTSIM, in which data for intact mooring lines and damaged mooring lines with different sea states were simulated. Inputs to the MLP were DGPS readings of the platform and the total mass of the FPSO platform. Backpropagation algorithm was used for training the MLP network, and the output of the MLP was a regression-based output, whereby the algorithm was able to predict which mooring line was damaged.

In [18], an MLP and an LSTM network to detected mooring line damage of an FPSO platform was implemented. Two approaches were used for identifying mooring failures: an auto-correlation approach where the future sway motion is predicted based on previous sway motion and a cross-correlation approach where five DoF (surge, heave, roll, pitch, and yaw) of an FPSO is used to predicted sway motion. Both MLP and LSTM models were able to detect mooring line failure. Information about the networks' configuration was not provided.

A convolutional neural network (CNN) based binary classifier was implemented for mooring line failure detection in [11]. A non-disconnectable turret FPSO with 21 mooring lines was modelled in `Orcaflex`[1] software. Orcaflex is a hydrodynamic analysis software, and it was used to make time-domain dynamic numerical simulations of the modeled FPSO vessel with its mooring lines. Environmental forces on the FPSO were accounted for in the numerical simulations by using Response Amplitude Operators and Quadratic Transfer Functions for accounting for the wave forces and drag force coefficients to account for the wind and current forces. Simulations for intact and single mooring line damage cases under different sea states were generated as the dataset for the proposed CNN.

The results from the generated simulations were encoded into label images. The in-house algorithm embedded information such as statistics of the horizontal position of the FPSO, Root Mean Squared values of the 6DoF of the vessel, and Root Mean Squared (RMS) values of the FPSO acceleration.

Evaluation of the proposed CNN performance was done with confusion matrix and F_score. The proposed CNN approach showed excellent results. However, there were cases of false negatives.

[1] https://www.orcina.com/orcaflex/

TABLE 1. Comparison between ML model and our proposal.

Paper (year)	ML technique	Inputs
[9] (2016)	MLP	Wave height, period and direction, wind velocity, current velocity, draft of the vessel
[15] (2017)	MLP	DGPS & Weather measurement & 6DoF motion sensors
[16] (2017)	MLP	DGPS & Platform mass
[17] (2018)	MLP	The long drift periods and mean values of the surge and sway motions of the vessel
[18] (2018)	MLP & LSTM	DGPS & 6DoF motion sensors
[11] (2019)	CNN	Statistics of the horizontal position parameters of the vessel and the root mean square values of the 6DoF accelerations
[3] (2020)	MLP	Mean and standard deviation environmental data & 6DoF motion sensors
Our proposal	MLP & LSTM	DGPS & 6DoF motion sensors

In [3], the authors used an MLP model with five hidden layers to detect damaged mooring line of a tension leg platform (TLP). Inputs to the MLP were the mean and standard deviation of wind and wave environmental measurements and the TLP floater response (6DoF: sway, yaw, surge, pitch, heave, roll). The output of the network was a binary classification of the mooring line status, that is, compromised or not compromised mooring system. The developed MLP was trained and tested on simulated data generated by CHARM3D software from Texas A&M University. During dataset generation, the TLP with intact mooring lines and a single mooring line damage was created. The MLP was trained with the TLP without a comprised mooring system and tested with a TLP response with a damaged mooring line. Random noise was added when data was being generated in order to replicate real-life weather conditions. The MLP model developed was able to detect when a mooring line damage occurred.

From these works, a common attribute could be seen. Input to their respective NN had either a combination of DGPS, sea state measurement, or floater response. Our proposed approach draws inspiration from the work of [18] by combining the auto-correlation approach and cross-correlation approach they implemented of the platform 6DoF. In their implementation, only RMS error was used to identify the occurrence of mooring line breakage. On the other hand, we used two more errors – RMSE, mean and median – to complement and affirm the occurrence of a break in the mooring line. In their implementation for both approaches – cross-correlation and autocorrelation – the LSTM model used a time interval of 1 second of input to predict the subsequent second, while the time interval window for the MLP model for both the approaches had two windows, 10 and 100, respectively. In their experiments, the authors found that the longer window decreased the MLP overall performance. Our network, on the other hand, has a long forecast window when compared to theirs. Our MLP model uses 600 seconds of the previous platform's motion to predict 100 seconds of the platform's future movement, while the LSTM model uses 1000 seconds of input to predict 400 seconds of the platform's future movement.

As we can see in the Table 1, in our proposal the platform motion response (6DoF motion sensors and DGPS) is provided as inputs to the networks. In addition, the forecast length of both our models is longer when compared to the other methods.

III. PRELIMINARIES

Machine learning (ML) is a subset of artificial intelligence, in which a machine learns from data without being issued explicit instruction on how to solve a given task [4]. This section presents the fundamentals of supervised learning that is a subcategory of ML used in this article. Supervised machine learning involves training a model to generate a mapping function that can predict output for a given input. The model trains on labelled data until it achieves a reasonable accuracy in making accurate predictions. In this article, ML is used in a supervised manner to make a time series prediction of an offshore platform motion. The supervised models used here are artificial neural networks (ANN).

A. NEURAL NETWORK PRINCIPLES

Most ML algorithms are based on ANN, implementable in different forms, although their primary function is always similar. ANN can change some parts of its structure based on received internal or external information. The essential ability to change its structure enables ANN to learn patterns in data structures. ANN is a parallel computational model consisting of adaptive processing units, known as neurons, that are densely connected to each other. ANNs are inspired by neurons and synapses in the human brain. ANNs are composed of neurons and connection lines, which are structured in layers to form a network. A basic ANN has an input, one or more hidden layers and an output layer [10]. A neuron unit is also known as perceptron, shown in Figure 2.

A single unit of perceptron takes in inputs $\mathbf{x}$, given by $x_1, x_2, x_3, .., x_n$, to generate an output $\hat{y}$. Each input (i) is connected to a node (j) with weights $\mathbf{w}$, with $w_1, w_2, w_3, .., w_n$ assigned to each input respectively. These weights determine the strength of a connection to the node (j). The node (j)

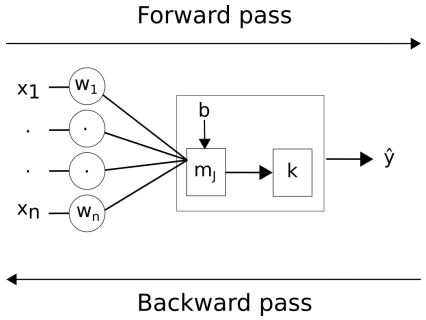


FIGURE 2. A single unit of a perceptron, where $\hat{y}$ is the output, k is the activation function, m_j is the summed weight, b is the bias, x is the input, and w is the weight.

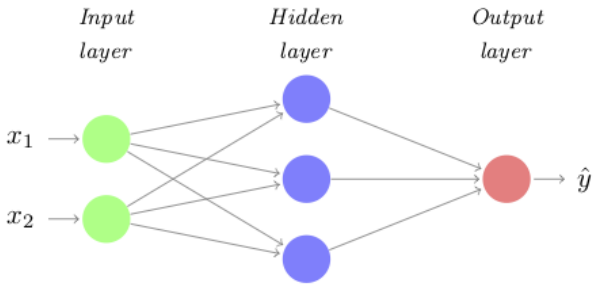


FIGURE 3. An MLP diagram.

generates a weighted sum of all the inputs values plus bias:

$$m_j = \left(\sum_{i=1}^{n} w_i.x_i + b \right), \tag{1}$$

where the n is the number of nodes in the input, x_i is the input value in node i, w_i is the weight in the connection from i to j, b is the bias. The result of the weighted sum (m_j) is passed through an **activation function** k to generate an output l_j

$$l_j = k(m_j), \tag{2}$$

where if l_j is an input to another perceptron, it becomes x or $\hat{y}$ if it is the final output. The activation function k can be referred to as a gate that standardizes information. There are different types of activation functions available. However, for brevity, the most common activation functions, sigmoid, and ReLU, are explained. Sigmoid activation function whose formula equals:

$$k(z) = \frac{1}{1 - e^{-z}} \tag{3}$$

is governed by the following: the value coming from weighted sum (here $z = m_j$) is squashed to be in the range of [0, 1], and it is propagated forward. ReLU whose formula equals $k(z) = \max_z(0, z)$ is governed by the following, if the weighted sum ($z = m_j$) is greater than zero the information from the input is allowed to pass while if $z = m_j$ is 0 or negative it is clipped at 0 and 0 is passed as output [7]. Depending on the activation function's decision, k the output $\hat{y}$ is gotten. An ANN learns by using the error difference between its output and the actual value to improve. The learning process is done in three steps. The first step involves the flow of information from input to the activation function to produce an output. This step is known as **forward pass**. In the second step, the output of the ANN $\hat{y}$ is compared against the actual value y, given that this is supervised learning where the actual output (also called label) is known beforehand in a training set consisting of pairs $< \mathbf{x}, y >$; this is known as **loss function calculation**. In the third step, the error difference is then back-propagated to adjust the weights of the ANN. Starting from the output layer back to the hidden(s) layer(s) and back to the input layer, this is known as **Backward pass**. These three steps are done recursively until a stopping criterion is met. The stopping criterion can either be allowing the network to iterate n number of **epochs** or when the error difference between the network predictions and the true label is minimal. Epoch refers to the number of times the network is instructed to cycle through the whole data set. These learning steps are what is referred to as backpropagation.

B. MULTI-LAYER PERCEPTRON

A **multi-layer perceptron (MLP)** is a feed-forward neural network that is created when multiple perceptrons are structured in layers to solve complex problems, as illustrated in Figure 3. In feed-forward networks, information propagates in one direction. Information moves from the input layer to the hidden layer to the output layer.

The MLP follows the single perceptron principle but on a larger scale. Perceptrons are structured in layers to construct an MLP network. These layers are the input layer, one or more hidden layers, and the output layer. Following the same principle of a perceptron unit, information flows from the input layer through the hidden layers, after which it passes to the output layer to make a prediction. Training is done using the **back-propagation algorithm** to improve the accuracy of an MLP. The back-propagation algorithm works by using the error difference between the MLP output and the expected output to make adjustments to the weights in each layer of the network, starting from the output layer backwards to the input layer with the objective of reducing the error difference of the MLP output and the actual output, the same way as explained before. A typical performance metric used to gauge the difference between the model prediction and the actual value is the **Mean Squared Error (MSE)**

$$Loss\ function = \mathsf{MSE} = \frac{1}{2}(\hat{y} - y)^2$$

which can be regarded as the loss function. The weights and bias of the network can be adjusted to reduce the loss function. The back-propagation algorithm employs different optimizers to reduce the loss function. The simplest of them is the gradient descent that finds the slope of the loss function using the partial derivative with respect to each weight in a layer, after which the weights and bias are updated until the loss function is minimal. Other frequently used optimizers

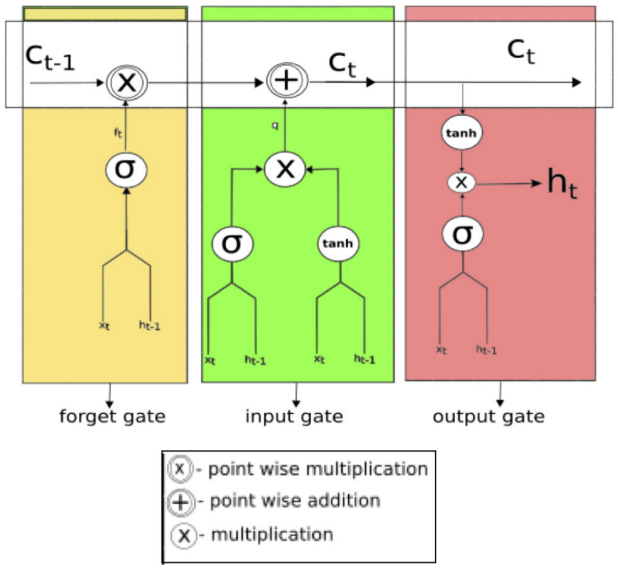


FIGURE 4. An LSTM diagram, where c_{t-1} is the previous cell state, c_t is the new cell state, h_t is the new hidden state, *ft* is the forget gate, σ is the sigmoid activation function, tanh is the hyperbolic tangent function, h_{t-1} is the previous hidden state.

are the stochastic gradient descent with momentum, adaptive gradient, and Adam optimizer, among others.

In the next section, a variant of neural network different from feed foreword network is introduced.

C. RECURRENT NEURAL NETWORKS

Recurrent neural networks (RNN) build upon the principles of feedforward networks, introducing a further loopback mechanism where information from previous time step $t-1$ is reused in the current time step t to produce an output. This mechanism, in turn, gives RNN the ability to retain information it has seen. By sharing information about the previous output between each node in each recurrent hidden layer, the RNN is able to learn the order of temporal data. However, it leads the RNN to face the vanishing gradient problem during training when the weights of the RNN network are being adjusted. This problem is due to the gradient descent method in the backpropagation algorithm. Each node in a layer calculates its gradient concerning the effect of the gradient in the previous layer. When the previous layer's adjustments are small, the weights of the nodes of the current layers will be even smaller when adjusted, making the gradients continuously smaller. Thus the vanishing gradient problem is encountered.

Long Short Term Memory (LSTM), a variant of RNN, was developed to solve the vanishing gradient problem. The RNN structure is improved upon by having three gates and a cell memory in each node. These gates are the input gate, forget gate, output gate, and cell state, which all together make up an LSTM unit, shown in Figure 4. Using these gates, the LSTM manipulates information flow through gates, storing information of relevance to the network within the network's memory.

LSTM is keeping the cell state by a belt-like structure. This cell state is shared by a LSTM cell equally and each cell can manipulate its cell state in 3 different manners, also known as gates. Outer events have no influence on the cell state. Gates are used to manipulate the cell state. They are activated by a sigmoid activation function and consist of a pointwise multiplication operation. Sigmoid layer can take a value between 0 and 1, where 0 means that the gate is closed and with 1 the influence of this gate is maximal.

The entrance gate of an LSTM unit is the forget gate (f_t). Its *sigmoid activation function* (σ) defines the amount of information that should be forgotten or kept in the cell state. The gate decides based on the previous output h_{t-1} and the input x_t. Its activation function squashes these two information to outputs 0 or 1, where 1 denotes retain the information and 0 denotes forget the information stored:

$$f_t = \sigma(w_{f_x}.x_t + w_{f_h}.h_{t-1} + b_f). \tag{4}$$

The input gate (i_t) of the LSTM cell decides the information to be allowed into the memory of the LSTM cell. Two copies of the current input (x_t) and prior state (h_{t-1}) are created where one copy goes to a sigmoid activation function (σ) whose value 1 means allowing information into the memory and 0 denies information flow into the LSTM cell. The second copy made goes through a *tanh activation function* whose output ($\tilde{c_t}$) is between -1 and 1, which is used to help regulate the LSTM cell. The product of the two activation functions are found and stored as q,

$$\begin{aligned} i_t &= \sigma(w_{i_x}.x_t + [w_{i_h}.h_{t-1}] + b_i), \\ \tilde{c_t} &= \tanh(w_c.[h_{t-1}, x_t] + b_c), \\ q &= i_t.\tilde{c_t}. \end{aligned} \tag{5}$$

The output gate (o_t) decides what information is stored in the memory of the LSTM cell by either choosing to preserve the information already in the LSTM memory or adding more information to the memory. The output is the new hidden state (h_t) of the LSTM unit, which is sent to the next LSTM in the hidden layer as (h_{t-1}). The output gate takes the new memory cell (c_t) information and passes it through a *tanh activation function* and also takes the prior hidden state (h_{t-1}) and the information of the current input (x_t) and passes it through a sigmoid activation function. The outputs of these two activation functions, sigmoid and tanh, are multiplied together to produce the new hidden state (h_t),

$$\begin{aligned} o_t &= \sigma_o(w_{o_x}.x_t + w_{o_h}.h_{t-1} + b_o), \\ p_t &= \tanh(w_o.[c_t + b_c), \\ h_t &= o_t.p_t. \end{aligned} \tag{6}$$

In the next section, we present our study that consisted of comparing an MLP model with an LSTM model to perform the prediction of a floating platform's motion.

IV. METHODS

The development of our failure detection system for mooring lines is aimed at a Floating Production Storage and

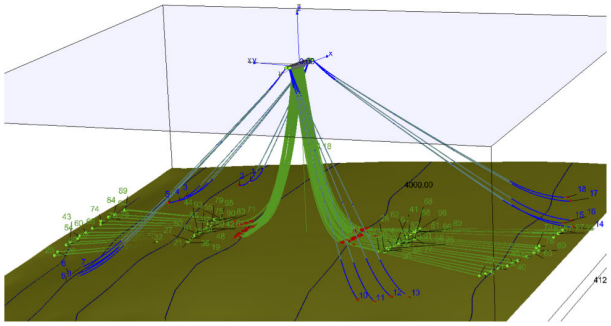


FIGURE 5. FPSO platform model with 18 anchor lines (with blue tips) and risers (in green).

TABLE 2. Platform dimension and setup.

Unit	Hull dimension
Length between prep. (m)	320
Length overall (m)	337
Lateral windage area (m²)	16800
Frontal windage area (m²)	2041
Draft (m)	16
Beam (m)	54.5
Depth (m)	27.8
Number of risers	79
Number of mooring lines	18

Offloading platform (FPSO). The platform used in this work is a converted very large crude carrier (VLCC) spread moored FPSO, with fixed heading. It has 18 mooring lines (8 at the bow, 10 at the stern) and 78 risers, as shown in Figure 5. In the experiments reported here, a fixed draft was used. The platform characteristics such as the mass, dimensions, number of risers and mooring lines can be found in Table 2.

A combination of data from a Differential Global Positioning System (DGPS) – which provide surge and sway position and yaw angle – and an inertial measurement unit (IMU) – which give roll and pitch angle and heave position, resulting in the variables that constitute the platform's 6 degrees of freedom (6DoF), is used as inputs to our proposed solution. The solution consists of estimating time series's prediction of the platform movement variables based on the observation of the past series. We here analyze the predictions made by an MLP predictor and an LSTM predictor, comparing the results of both in several aspects.

Both predictors are trained in a supervised manner to estimate future movement with data from the platform with intact mooring lines. The hypothesis is that, when there is a failure in the mooring lines, there is a significant difference between the predicted value and the value measured by the DGPS and IMU sensors, which makes the system able to notify a probable line failure. For this, the system calculates the error between the estimated and measured values. Error values above a predefined threshold indicates whether or not there was a failure in the platform's mooring line.

Errors are calculated within error windows, with the size of two prediction time intervals. The stride of the error windows

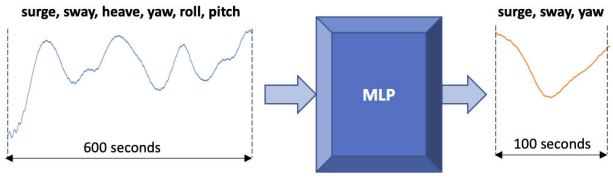


FIGURE 6. The MLP model has as input 600 seconds of time series of sway, surge, heave, roll, pitch, and yaw of the platform motion, and outputs 100 seconds of prediction of sway, surge, and yaw.

is one prediction window. For each error window the mean error (ME), median error (MedE), and the root mean squared error (RMSE) were calculated,

$$\begin{aligned} ME &= \frac{\sum_{i=0}^{n} \Delta_i}{n}, \\ MedE &= V_{sorted}[(N-1)/2], \\ RMSE &= \sqrt{\frac{\sum_{i=0}^{n} \Delta_i^2}{n}}, \end{aligned} \tag{7}$$

where n is the number of time steps in the error window, Δ is the point-to-point difference between the predicted and the simulated data, V_{sorted} represents a sorted array of the Δ values and N represents the number of array elements. In the following sections, each of the ANN models is detailed.

A. MLP ARCHITECTURE

The best MLP model capable of learning the platform's complex movements was defined experimentally, after some tests with different models with different layer compositions, activation functions, inputs, and outputs. The MLP input consists of 600-second period in time series for each of the 6DoF (sway, surge, heave, roll, pitch, and yaw), and the output is 100 seconds, with 1-second time step, of the predicted time series of 3 DoF (surge, sway, and yaw). Figure 6 shows input and output of the MLP model.

The implemented MLP model comprises of an input layer, three hidden layers, and an output layer, with each node in a layer fully connected to each node in the next layer. An illustration of the MLP architecture can be seen in Figure 7. The MLP has 3,600 input nodes (corresponding to 600 × 6) and has 300 output nodes (corresponding to 100 × 3). The three hidden layers have 7200, 3600 and 1800 nodes respectively in each layer. The ReLU activation function is used in all the layers except for the output layer, which uses linear activation function to make the prediction. Adam optimizer algorithm with a learning rate of 1-e7 was used to optimize the MLP network.

B. LSTM ARCHITECTURE

The implemented LSTM model predicts 400 seconds based on 1000 seconds of horizontal platform motion, as shown in Figure 8.

The LSTM model used is an encoder-decoder model with two LSTM layers. The first LSTM layer is trained in understanding the input sequence. It encodes the input sequence

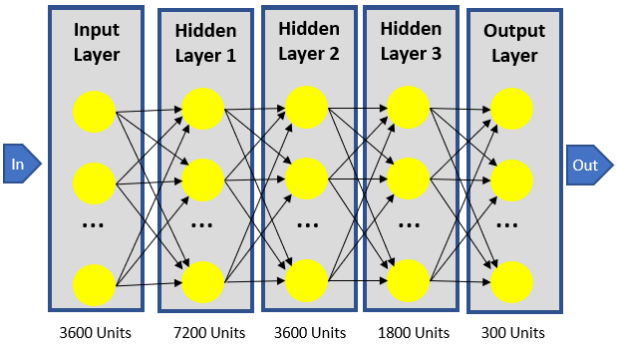


FIGURE 7. The fully connected Multi-Layer Perceptron (MLP) architecture used, with 3600 input nodes, 300 output nodes, and three hidden layers with 7200, 3600, and 1800 nodes, respectively.

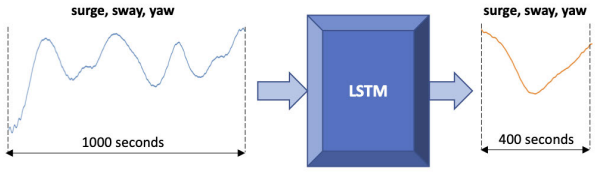


FIGURE 8. The LSTM model using the last 1000 seconds of the features surge, sway and yaw to predict 400 seconds of these three features.

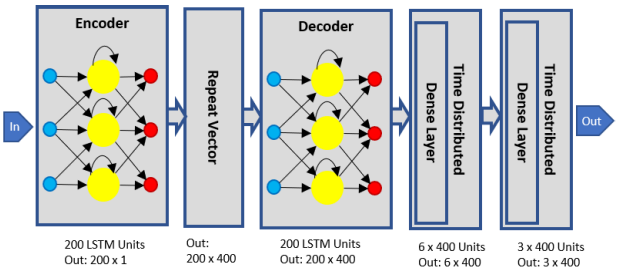


FIGURE 9. The encoder-decoder LSTM architecture used with one repeat vector layer and two time distributed layer.

into a vector with a fixed length, which is then interpreted by the decoder. The decoder creates an output sequence based on the encoded information. The encoder-decoder model is commonly used for sequence to sequence problems [2]. Since our problem is also a sequence to sequence problem, the given architecture is expected to achieve good performance. There are two layers between the encoder-decoder model, a repeat vector layer and a time-distributed layer, respectively, as illustrated in Figure 9.

The first LSTM Layer consists of 200 units. Each unit of the layer responds to the seen input, which means that the output of this layer is a vector of 200 values. This layer is considered the encoder, and it is trained to understand the input and translate it into a fixed-length vector. Since the decoder LSTM layer needs a two-dimensional input, a repeat vector layer is needed. As the output of the predictor are 400 steps, the input is repeated 400 times.

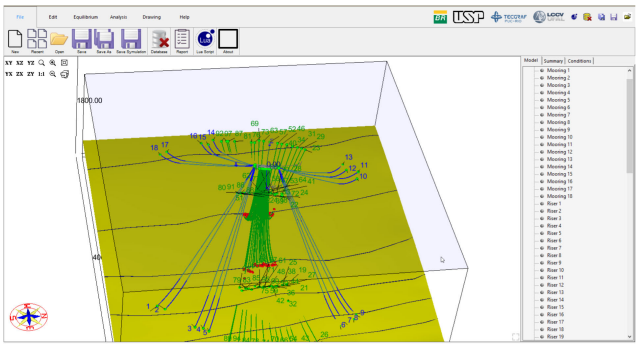


FIGURE 10. Dynasim Interface (source: Dynasim User Manual).

The second LSTM layer can be understood as the decoder of the structure. It also consists also of 200 units, and each unit gives a response based on the encoded message. This layer returns the hidden state for each input time step creating a two-dimensional output. After the internal vector is decoded, it needs to be translated into the needed output form. This output is accomplished by two dense layers that reduce the 200 output values to the desired 3 DoF motion variables representing the horizontal platform motion (surge, sway, and yaw). Since dense layers work with one-dimensional inputs, they are wrapped in a time distributed layer. All layers use ReLU as an activation function and stochastic gradient descent (SGD) optimizer algorithm to optimize the network.

The proposed ANN predictors were implemented and trained with simulated data. In the following sections, we describe the experimental procedures and the results achieved. The performance of the two models, MLP and LSTM, were analyzed and compared.

V. EXPERIMENTAL SETUP

The analysis of the proposed ANN models' performance for motion prediction was made using simulation data to create motions of an FPSO platform with 18 mooring cables and 78 risers (see Table 2), under various environmental conditions.

The simulator used is Dynasim that is a numerical simulator developed and maintained by Universidade de São Paulo (USP)[2] and Petrobras,[3] among others [13]. The simulator allows the study of moored platforms' dynamic behavior and the analysis of offloading with a dynamic positioning system. Dynasim uses a floating platform's specifications together with the environmental conditions (wind, current and 2 components of waves), to produce time series of the platform's position, speed and acceleration (considering the platform's 6DoF). Figure 10 shows the Dynasim interface.

Figure 11 illustrates the general pipeline for data preparation, both for supervised training and for testing the two ANN models.

[2]http://tpn.usp.br/
[3]https://petrobras.com.br/en/

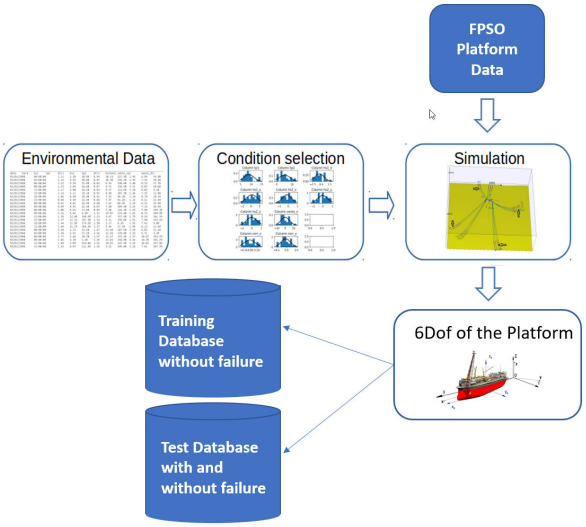


FIGURE 11. Pipeline of the platform motion prediction approach to detect platform line failures.

A. DATASET GENERATION PHASE

Real data were initially collected on the environmental conditions of the region of interest. These environmental conditions were retrieved from a weather station located in Campos Basin (Bacia de Campos) of Rio de Janeiro, Brazil, associated with the hydrodynamic models from 2003 to 2006 in intervals of 3 hours, totaling 18000 different environmental conditions, composed by the two components of waves which are characterized by their height (hs), peak to peak time (tp), and direction (dir). Wind and currents are characterized by their speed (vel) and direction (dir). In order to avoid using angles to indicate directions, the speed was divided into its x-component and y-component,

$$x = v * \cos(\phi); \quad y = v * \sin(\phi), \tag{8}$$

where v is the velocity and ϕ is its direction. For the wave and the swell, their angles are multiplied by the height. Whenever pertinent, these data are standardized for values between 0 and 1.

Table 3 illustrates a sample of the scaled environmental conditions gotten from the weather station located in Campos Basin with the angles of the current and wind measurement decomposed in x and y components.

Then, these environmental conditions data were sampled and fed into the Dynasim simulator, which, together with the data related to the platform used (see Table 2), generates the motion data of the platform without failures in the mooring lines for the training of predictors. The environmental data are randomly sampled from the real data, aiming that the generated samples follow the same distribution as the collected real data. Table 4 illustrates the result of this sampling, showing the mean values and respective standard deviations of the real data and the sampled data that were fed to the simulator for comparison purposes.

B. TRAINING PHASE

After sampling the environmental cases that best represent the region's environmental conditions, the next step was to make the simulated data suitable for training ML algorithms. As the ML algorithms, we follow the supervised learning paradigm, training sets must be created, consisting of a set of pairs of input and desired output.

For each of the 5000 selected cases of environmental conditions for training, Dynasim was used to simulate 3 hours of platform motion. We sample parts of the time series that characterize the simulator's platform motion for each environmental condition. Each of these parts represents a data window and corresponds to a training unit. The size of the data window is a system parameter and is kept fixed for each ANN model. Data windows are independent of each other and can be drawn from completely random points within this three-hour data regarding the same environmental condition. Likewise, data windows are sampled from other selected environmental conditions, and the set of all these training units form a training set for a neural network. Depending on the ANN model to be trained, the size of these data windows and the 6DoF motion variables (here called features) can vary.

A training unit is composed of the pair $< input - -output >$, which is defined by a partition of the data window's time series. For the MLP model, the training unit consists of input of 600s and an output of 100s, forming a data window of 700s. For the LSTM model, the training unit consists of input of 1000s and an output of 400s, forming a data window of 1400s.

The process of generating training units consists of sliding the data window on the time series generated by Dynasim, with strides defined as system parameters. It is essential to notice that data windows can overlap.

After the simulated data is split into several training units, they are used to train the ANN model. A good training data set is critical to the success of the models' ability to learn. The MLP network was trained for 3000 epochs using 5000 environmental conditions and another 1000 environmental conditions for validation. A total of $55,0000$ training units and $11,0000$ validation units. The LSTM model used 1000 environmental conditions for training and another 200 environmental conditions for validation. This process totaled $10,0000$ training units and 2000 validation units. The LSTM model was trained for 1500 epochs. During training, the early stopping method from Keras[4] API was used to prevent both ANN from over-fitting. The early stopping method uses the training set and validation set to keep track of the network performance, and it stops the training process when the model stops improving on a validation set.

After training, the predictor model is used to predict platform motion in a testing phase.

[4]https://keras.io/

TABLE 3. Sample of environmental conditions scaled between 0 and 1.

tp1 (seconds)	tp2 (seconds)	hs1_x (meters)	hs1_y (meters)	hs2_x (meters)	hs2_y (meters)	wind_x (m/s)	wind_y (m/s)	current_x (m/s)	current_y (m/s)
0.19	0.16	0.53	0.62	0.42	0.46	0.40	0.41	0.44	0.56
0.22	0.24	0.59	0.43	0.47	0.29	0.30	0.52	0.56	0.54
...	...	...	...	...	...	...	...	...	...
0.21	0.34	0.40	0.24	0.38	0.53	0.89	0.58	0.74	0.35
0.21	0.00	0.52	0.25	0.52	0.48	0.31	0.29	0.55	0.23

TABLE 4. Mean values and standard deviations of variables for all real environmental conditions and sampled cases.

tp1 (seconds)	tp2 (seconds)	hs1_x (meters)	hs1_y (meters)	hs2_x (meters)	hs2_y (meters)	wind_x (m/s)	wind_y (m/s)	current_x (m/s)	current_y (m/s)
Mean value and standard deviations of each column for all real cases									
8.65	4.99	-0.89	0.35	-0.37	0.07	-3.25	-2.56	-0.07	-0.32
2.62	4.09	1.41	1.41	0.64	0.64	5.69	5.69	0.28	0.28
Mean value and standard deviations of each column for selected cases									
8.29	5.46	-0.67	0.27	-0.32	0.08	-3.18	-2.40	-0.06	-0.17
1.84	3.03	1.30	1.26	0.57	0.61	6.58	5.02	0.30	0.31

C. TESTING PHASE

The test data-set feeds the trained predictor model, which then makes predictions. The test units must have the same dimensions defined for the training units, i.e., the same input and output sizes must be used in the training and test phases. For each environmental condition, 10 testing units were used.

As simulated data is used here and, thus, there is control of the line breaks of the mooring lines, the idea is to provide input data and compare the ANN models' output data with the simulated data, measuring the disparity between them. Thus, sequential predictions are made using a test window in the motion time series that continually feeds the ANN model. Then the test window is moved, and a new forecast is made. The stride of the prediction window is never bigger than the prediction time range, i. e. the output size. Therefore there always exists a predicted data for every time step. The stride size is a system parameter. If the stride is smaller than the prediction time range, there can exists multiple predictions for one time step. In this case the mean value is calculated between all available previsions of that time step, resulting in a single prediction for every time step.

After this, the predicted platform motion can be plotted against the actual platform motion. The difference, given by

$$\Delta = Y_{pred} - Y_{sim}. \tag{9}$$

between the simulated, Y_{sim} and predicted motion Y_{pred} is then used to calculate the errors of the prediction.

To calculate the errors, we split the difference graphic into windows, hereafter referred to as difference windows, with the size of two prediction time intervals, which, as already mentioned, is a parameter of the system and depends on the ANN model used and the training done. The stride of the difference windows is one prediction time interval. The difference window size here is two times bigger as the prediction interval size, and the windows overlap. For example, for a predictor predicting 400 seconds, the difference window would have a size of 800 seconds, and its stride would be 400 seconds, as it can be seen in Figure 12.

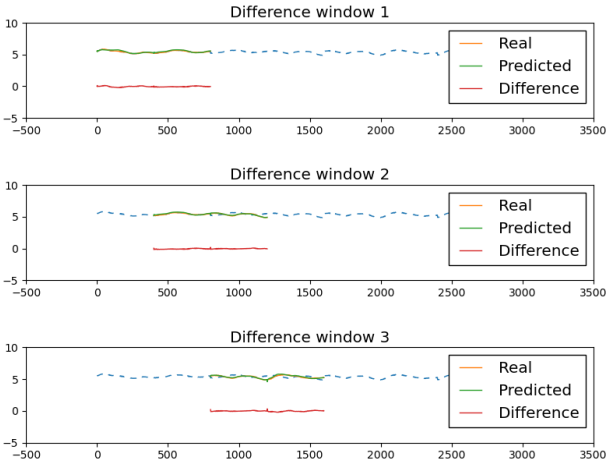


FIGURE 12. First three difference windows. Each graphic shows two predictions of 400 seconds concatenated in a 800s difference window, and the respective difference (in red) between the predictions (in green) and the actual data (dashed in blue). We can observe the 400s stride of the difference window that occurs between one graphic and another.

We use different error measures for each difference window: ME, MedE, and RMSE errors (see Eq. 7). We can plot the errors against the predicted and simulated platform motion as shown in Fig. 13, in which the errors are sequentially concatenated for visualization purposes. The three errors are calculated for each difference window.

FIGURE 13. Error Calculation. Top: predicted (orange) and actual (blue) motion values. Bottom: zoomed window with difference motion values (green).

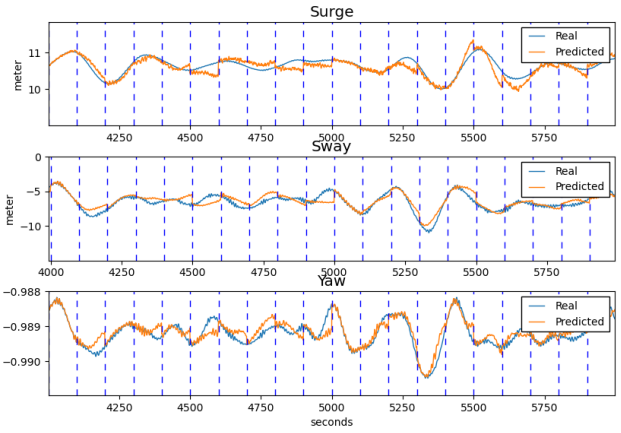


FIGURE 14. Illustration of MLP prediction on a single environmental condition with all mooring lines intact. The orange line is the MLP prediction and the blue line is the simulated platform motion. Top: surge, middle: sway, and bottom: yaw.

The last step is to detect if there was a line break within that test range. As the network was only trained without line breakages, the difference between the simulated platform motion and the predicted platform motion is expected to be higher for cases with line breakage. Results are presented in the next section.

VI. RESULTS AND DISCUSSION

The Figures shown in this section are the results of the tests carried out to assess the performance of the two models implemented. In each subsection, we detail the test data used, and we elaborate on the result and discussion obtained.

A. PREDICTION OF INTACT MOORING LINE

We began our test phase by assessing both ANN models' prediction accuracy on platform motions without mooring line breakage. Both model were given the same platform motion response with mild environmental conditions (see Table 7) to predict. Figure 14 for the MLP model and Figure 15 for the LSTM model shows both model prediction on platform motion without mooring line failure.

As can be seen in these figures, both models could predict the platforms' motions for the test data given to them. The results showed that the LSTM model performed better at predicting the platform motions than the MLP predictor. These results were confirmed by comparing the error scores of both models for the same test data shown in Table 5. A comparison between the MLP and LSTM error score for all test scenarios showed that the LSTM errors were always smaller than the MLP error score in all test scenarios. An explanation for this may be due to the nature of the LSTM model. A static ANN model is made up of algebraic equations only (for example, a feed-forward ANN, class to which MLP belongs). On the other hand, a dynamic model obeys differential equations (or partial differentials), where the variable is the time, and possibly algebraic equations as well (for example, RNN, class

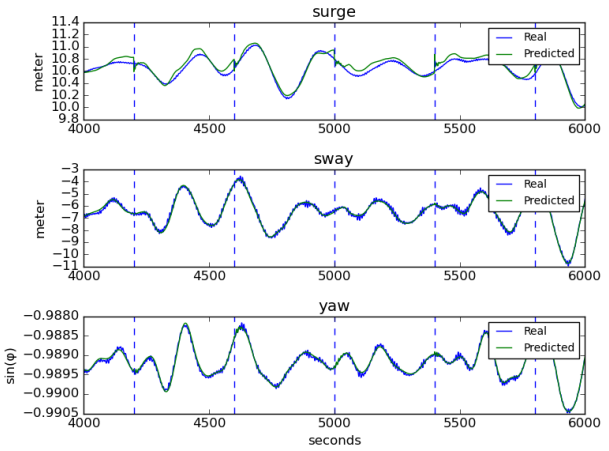


FIGURE 15. Illustration of LSTM prediction on a single environmental condition with all mooring lines intact. The simulated data is in blue and the predicted in green. Top: surge, middle: sway, and bottom: yaw.

to which LSTM belongs) [5]. Thus, dynamic model allows for the persistence of information, better dealing with the series's temporal and noisy aspects, which is the case with our application.

B. PREDICTION OF PLATFORM MOTION WITH MOORING LINE BREAKAGE

Next, the trained ANN models were then given platform motion with mooring line breakage to predict, under the same environmental conditions tested with all mooring lines intact.

Figure 16 shows the MLP network predictions for case when there is a mooring line breakage. It can be seen that after breakage of a mooring line, at approximately 5000 seconds, an offset in the platform position occurs, and the MLP model is unable to predict the motions of the platform henceforth.

TABLE 5. RMSE, Mean and Median error scores of the MLP and LSTM model on platform motion with intact mooring lines.

ANN Model	Surge (Meters)			Sway (Meters)			Yaw (Degrees)		
	RMSE	Mean	Median	RMSE	Mean	Median	RMSE	Mean	Median
MLP	0.300	-0.237	-0.264	0.619	-0.540	-0.509	2.337 e-04	-1.311 e-04	-1.007 e-04
LSTM	0.125	-0.105	-0.110	0.191	0.049	0.041	1.088 e-04	-2.254 e-05	-2.440 e-05

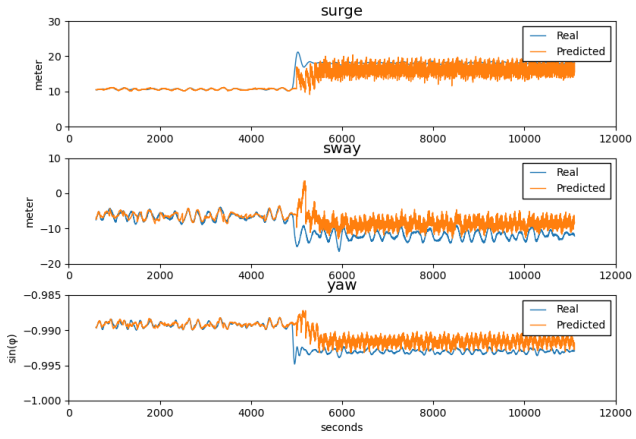


FIGURE 16. Illustration of breakage in mooring line L1 at approximately time step 5000. The orange line is the MLP prediction and the blue line is the simulated platform motion. Top: surge, middle: sway, and bottom: yaw motion of the platform.

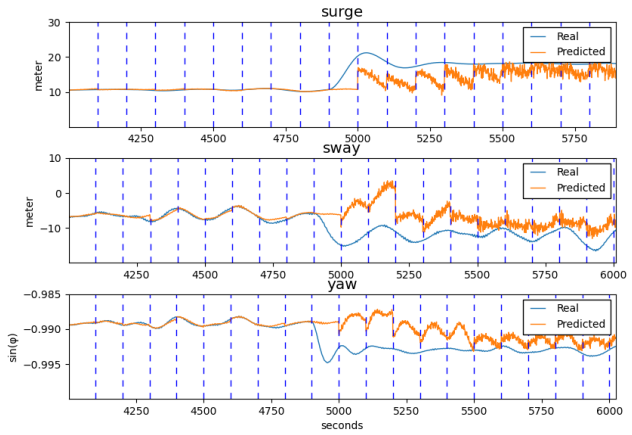


FIGURE 17. A zoomed illustration of breakage in mooring line L1. The orange line is the MLP prediction and the blue line is the simulated platform motion. Top: surge, middle: sway, and bottom: yaw motion of the platform.

Figure 17 shows a zoomed image of Figure 16 and it shows how the MLP prediction and the simulated platform motion deviates after a line breakage occurs after time step 5000s.

It shows for the surge feature, after line breakage, a change of −9 meters from the initial 15 meter happens. Surge measurement oscillates for three prediction window demarcated as the blue vertical lines, after which it stabilizes. The sway also shows change after a line is broken. An offset is also present between the simulated motion and the model prediction of 3 meters. The yaw feature also presents an offset after

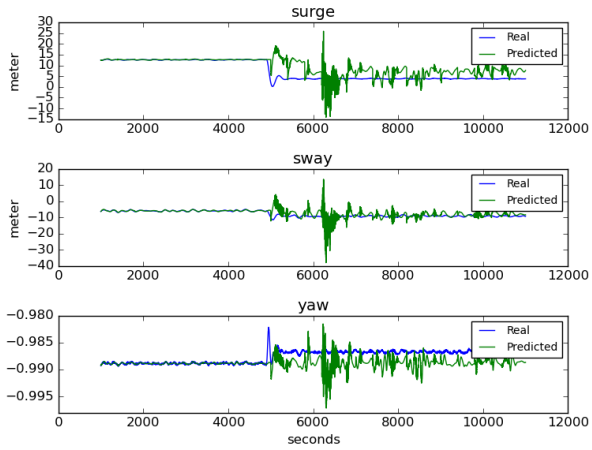


FIGURE 18. Illustration of breakage in mooring line L1 at approximately time step 5000. The green line is the LSTM prediction and the blue line is the simulated platform motion. Top: surge, middle: sway, and bottom: yaw motion of the platform.

line breakage, and the MLP model is no longer able to predict the motion.

Figure 18 shows the LSTM model prediction on the same environmental condition that can be seen in Fig. 15, with a simulated breakage in the Mooring Lines after 5000 seconds. The breakage is visible as a change in offset. After the breakage, it can be observed that the LSTM model has difficulties in predicting the platform motion. In the zoomed image version in Fig. 19, it can be seen a clear difference between simulation and prediction at the point in time of line breakage, since the platform changes its local position after the failure of the line and the predictor does not predict this change in position. Figure 19 shows two prediction windows during the period of line breakage. The predictor predicts the motion staying close to the predicted position while the simulated platform position changes 3 meters in surge and sway from its old position. It can be observed that the predictor has problems predicting the platform motion in the new location, not predicting the platform motion precisely anymore.

As can be seen in these figures, both models were unable to predict the platform motion by the test data provided to them after the mooring line breakage. The MLP model was unable to predict the platform motions accurately, and the LSTM model also demonstrated this difficulty in predicting the platform motion after mooring line failure but to a lesser extent when compared to the MLP network on the same platform motion. This was also confirmed by comparing the error scores on Table 6 of both models on the same platform

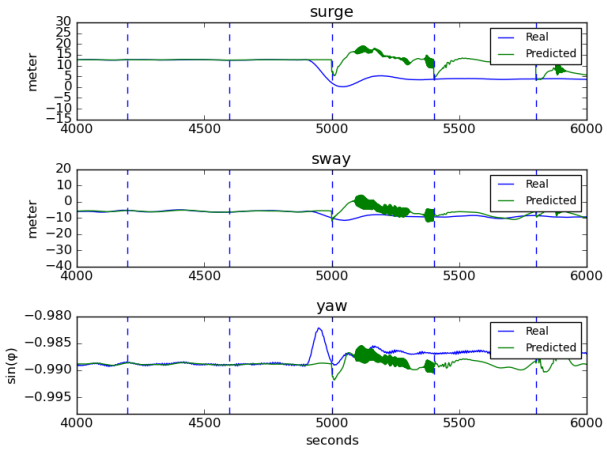


FIGURE 19. A zoomed illustration of breakage in mooring line L1. The green line is the LSTM prediction, and the blue line is the simulated platform motion. Top: surge, middle: sway, and bottom: yaw motion of the platform.

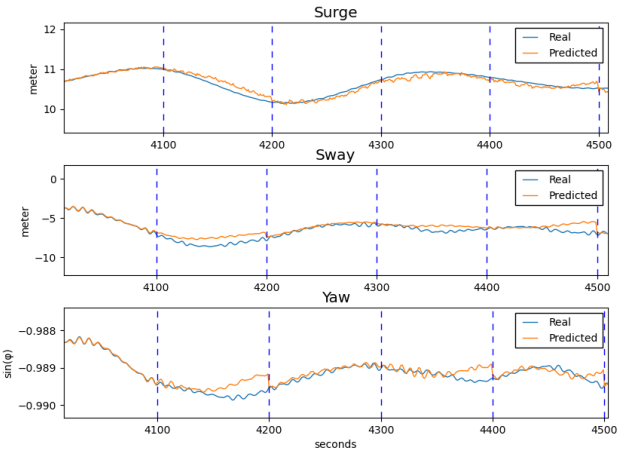


FIGURE 20. A zoomed illustration of MLP prediction with a mild environmental condition. The orange line is the MLP prediction, and the blue line is the simulated platform motion. The blue vertical lines in the figure mark the boundaries of a prediction window of the MLP model. Top: surge, middle: sway, and bottom: yaw motion of the platform.

motion data. This behaviour can be attributed to the fact that both models were trained on platform motions with intact mooring lines. So, after mooring line failure, the platform's motions with a compromised mooring system are unknown to these trained models.

C. PLATFORM MOTION PREDICTION ON DIFFERENT ENVIRONMENTAL CONDITIONS

The two predictor models developed were tested in various platform motions under different environmental conditions. For illustration, two different scenarios whose environmental conditions have been classified as mild and stormy are presented in Table 7. As can be seen, wave ($hs1$), swell ($hs2$) and wind speed ($wind\_vel$) in these scenarios are different. The prediction performance of the networks in the platform motion under these two scenarios is presented below.

Figure 20 shows the MLP prediction accuracy on the mild environmental condition, and the result shows the MLP model was able to predict the frequency and amplitude of the platform motions.

Figure 21 shows the MLP prediction accuracy on stormy environmental condition from Table 7. The result shows that the MLP model was unable to fully predict the oscillation of the platform's surge, sway, and yaw.

Figure 22 shows the LSTM prediction on the mild environmental condition, and the result shows the LSTM model was able to predict the frequency and amplitude of the platform motions.

Figure 23 shows the LSTM prediction of the accuracy of stormy environmental condition, and the result shows the LSTM model was unable to predict the frequency and amplitude of the platform motions.

It can be seen both models were able to predict the oscillation of the simulated platform in mild environmental conditions, but for a stormy environmental condition,

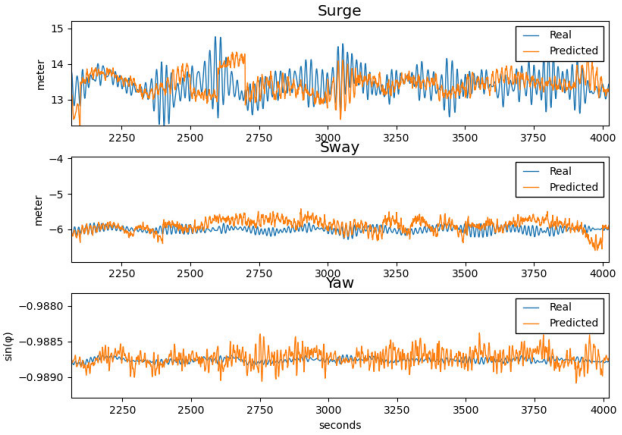


FIGURE 21. Illustration of MLP prediction on a stormy environmental condition with all mooring lines intact. The orange line is the MLP prediction, and the blue line is the simulated platform motion. Top: surge, middle: sway, and bottom: yaw motion of the platform.

the MLP could not predict the rapid motions of the platform while LSTM also found it difficult to predict the motion of the platform in environmental conditions with rapid oscillations, predicting only the slow, most influential components of the motion in these conditions. The LSTM performed better in all cases when compared against the MLP model.

A possible reason why both models found it challenging to predict the platform motion of environmental condition with rapid oscillation could be because the models were trained with only a few environmental conditions of these types in their training dataset. In the entire 18000 environmental condition measurements gotten from the weather station located in Campos Basin (Bacia de Campos) of Rio de Janeiro (RJ), Brazil, since 2003, there were only 100 environmental conditions with wave heights greater than 4 meters.

TABLE 6. RMSE, Mean and Median error scores of the MLP and LSTM model on platform motion with a broken mooring line.

ANN Model	Surge (Meters)			Sway (Meters)			Yaw (Degrees)		
	RMSE	Mean	Median	RMSE	Mean	Median	RMSE	Mean	Median
MLP	6.061	-5.296	-5.415	6.289	-6.086	-6.225	3.594 e-03	3.097 e-03	3.143 e-03
LSTM	8.306	-7.241	-7.666	3.928	-2.175	-2.177	2.995 e-03	2.956 e-03	2.817 e-03

TABLE 7. Two environmental conditions selected.

Sea state	Index	hs1 (meters)	tp1 (seconds)	dir1 (deg)	hs2 (meters)	tp2 (seconds)	dir2 (deg)	wind_vel (m/s)	wind_dir (deg)	current_vel (m/s)	current_dir (deg)
Mild	600	1.68	8.48	47.2	1.33	5.4	191.9	8.21	195.3	0.2	288.84
Stormy	8818	2.84	17.04	192.2	0.0	0.0	0.0	7.77	208.6	0.45	216.93

TABLE 8. RMSE, Mean and Median error scores for the MLP model on different platform motions and configuration.

case	Sea state	Surge (Meters)			Sway (Meters)			Yaw (Degrees)		
		RMSE	Mean	Median	RMSE	Mean	Median	RMSE	Mean	Median
Case 600	Mild	0.300	-0.237	-0.264	0.619	-0.540	-0.509	2.337 e-04	-1.311 e-04	-1.007 e-04
Case 8818	Stormy	0.748	-0.438	-0.472	0.579	-0.555	-0.546	2.066 e-04	-1.255 e-04	-1.288 e-04
Failure Case 600	Mild	6.061	-5.296	-5.415	6.289	-6.086	-6.225	3.594 e-03	3.097 e-03	3.143 e-03

TABLE 9. RMSE, Mean and Median error scores for the LSTM model on different platform motions and configuration.

case	Sea state	Surge (Meters)			Sway (Meters)			Yaw (Degrees)		
		RMSE	Mean	Median	RMSE	Mean	Median	RMSE	Mean	Median
Case 600	Mild	0.125	-0.105	-0.110	0.191	0.049	0.041	1.088 e-04	-2.254 e-05	-2.440 e-05
Case 8818	Stormy	0.450	-0.129	-0.151	0.163	-0.555	0.056	5.385 e-05	-2.165 e-05	-2.166 e-05
Failure Case 600	Mild	8.306	-7.241	-7.666	3.928	-2.175	-2.177	2.995 e-03	2.956 e-03	2.817 e-03

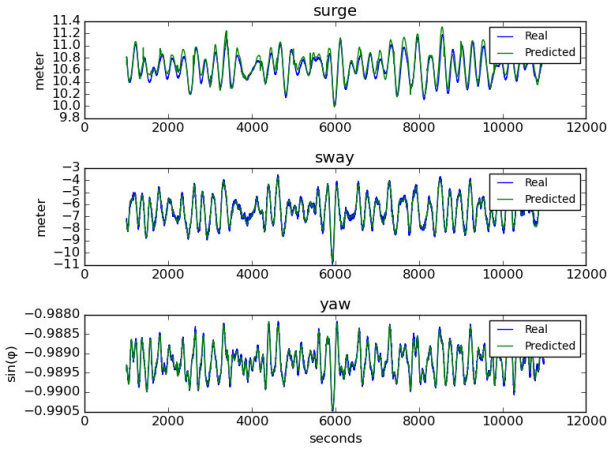


FIGURE 22. Illustration of LSTM prediction on mild environmental condition with all mooring lines intact. The green line is the LSTM prediction, and the blue line is the simulated platform motion. Top: surge, middle: sway, and bottom: yaw motion of the platform.

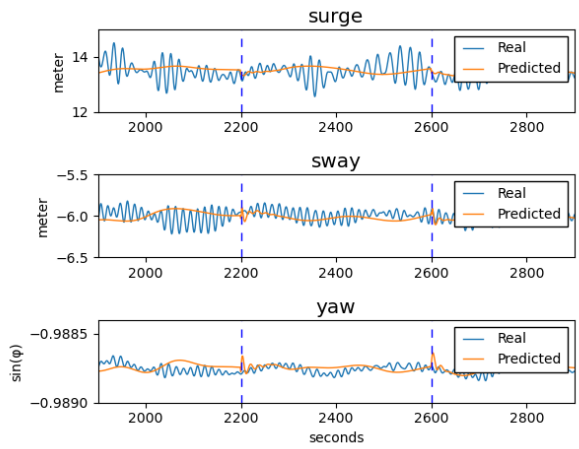


FIGURE 23. Illustration of LSTM prediction on stormy environmental condition with all mooring lines intact. The orange line is the LSTM prediction, and the blue line is the simulated platform motion. The blue vertical lines in the figure mark the boundaries of a prediction window of the LSTM model. Top: surge, middle: sway, and bottom: yaw motion of the platform.

D. ERROR CALCULATION

In this section, we present the error scores for different environmental conditions the MLP and LSTM models were tested on, and we show how these errors change concerning to the environmental conditions and mooring line status. Using the error scores of three different environmental conditions with different mooring line setups, the RMSE, mean and median errors of these conditions are compared.

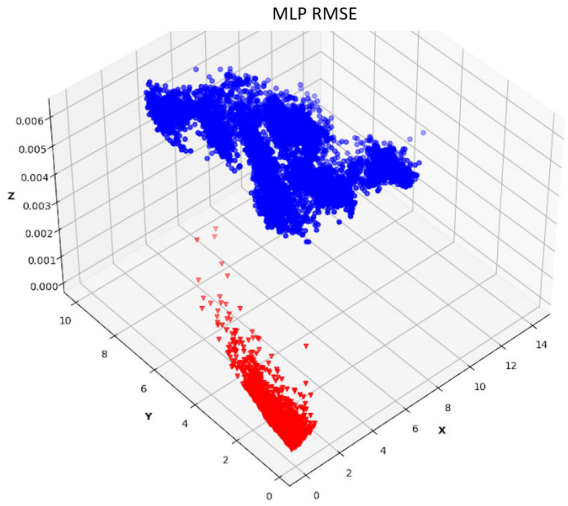


FIGURE 24. RMSE error score of features surge (in the x axis), sway (in the y axis), and yaw (in the z axis), for all environmental conditions in the test set. The red circles represent cases without mooring line failure, and the blue circles represent cases with a mooring line failure.

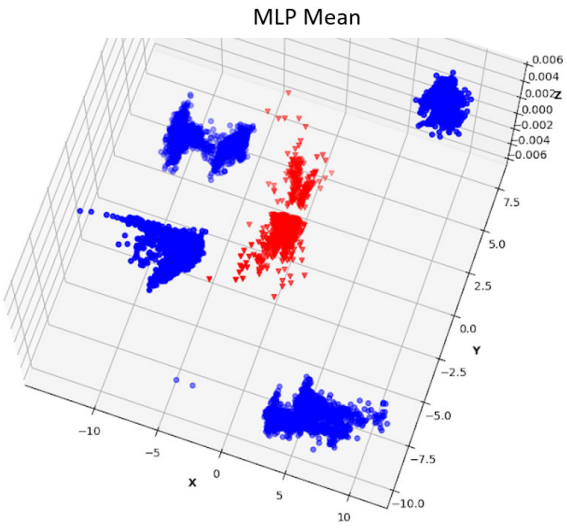


FIGURE 25. Mean error score of features surge (in the x axis), sway (in the y axis), and yaw (in the z axis), for all environmental conditions in the test set. The red circles represent cases without mooring line failure, and the blue circles represent cases with mooring line failures.

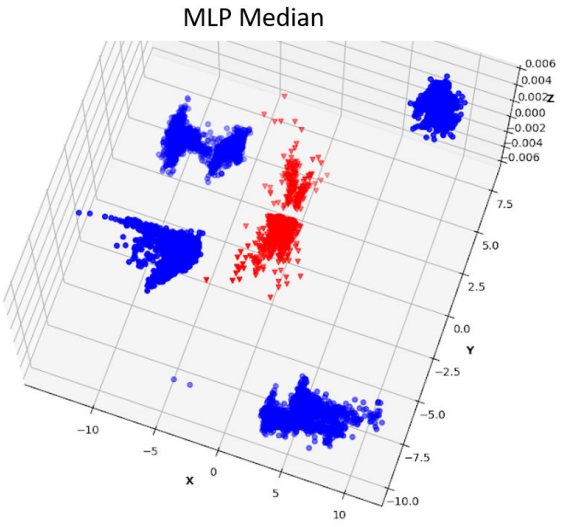


FIGURE 26. Median error score of features surge (in the x axis), sway (in the y axis), and yaw (in the z axis), for all environmental conditions in the test set. The red circles represent cases without mooring line failure, and the blue circles represent cases with mooring line failures.

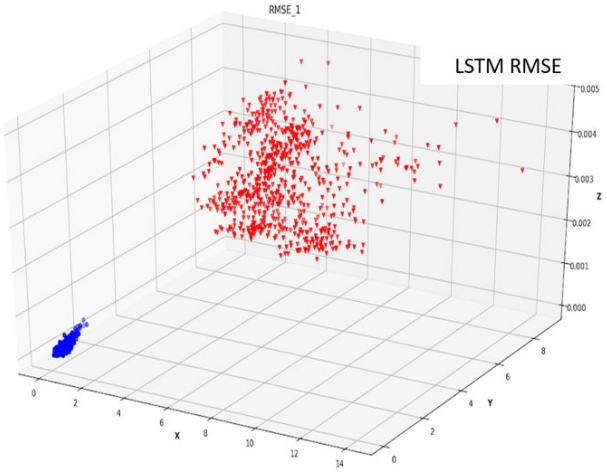


FIGURE 27. RMSE error score of features; surge (in the x axis), sway(in the y axis), and yaw (in the z axis) indexes for all environmental conditions in the test set. The red circles represent cases without mooring line failure, and the blue circles represent cases with mooring line failures.

The error score from the MLP model in Tab. 8 compares the RMSE, mean, median errors of platform motions on three environmental conditions used in the testing phase of this study. It can be seen when we compare the surge RMSE of case 600 – which has all its mooring lines intact – with the same case 600, but with a compromised mooring line, the error becomes ten times bigger. The same difference is seen for sway and yaw. There is always a difference in error score between situations with intact and comprised mooring lines under a specific environmental condition. The mean and median errors of these three environmental conditions also illustrate this fact for these comparisons. It can be seen a clear difference between the broken and not broken mooring line as it was expected.

The error score for the LSTM model in Table 9 also compares the RMSE, mean, and median errors of platform motions on three environmental conditions. The shown cases

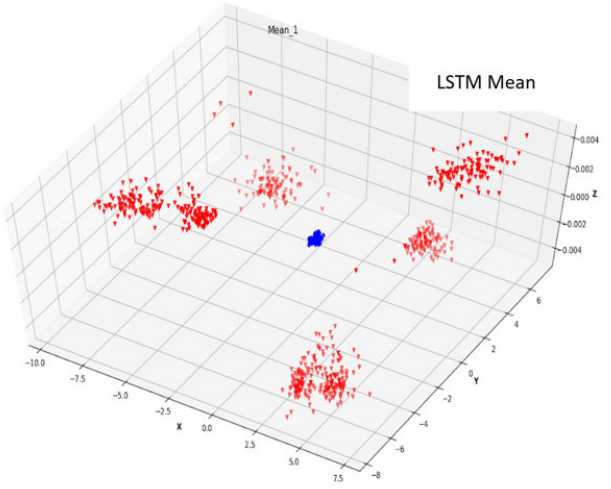


FIGURE 28. Mean error score of features; surge (in the x axis), sway(in the y axis), and yaw (in the z axis) indexes for all environmental conditions in the test set. The red circles represent cases without mooring line failure, and the blue circles represent cases with mooring line failures.

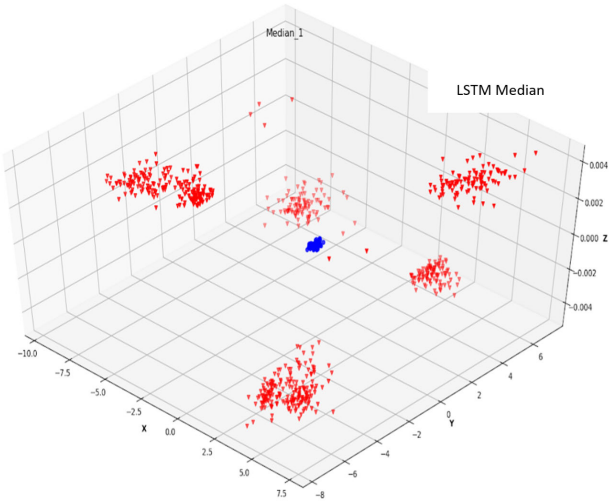


FIGURE 29. Median error score of features; surge (in the x axis), sway (in the y axis), and yaw (in the z axis) indexes for all environmental conditions in the test set. The red circles represent cases without mooring line failure, and the blue circles represent cases with mooring line failures.

are the selected environmental cases from Section VI-C, where case 8818 represents a stormy environmental condition and 600 a mild situation. Since RMSE always calculates the squared error, there can be no statement made over the error sign. It can be seen that there is a big gap between the scenarios with and without line breakage. It can be observed that the three error scores after a line breakage are higher than the case without line breakage.

Three-dimensional scatter plots are shown to visualize better how the errors are distinct from each other in different cases of environmental conditions with line breaks and without line breaks. Each axis in the plot corresponds to one of the features, surge (x), sway (y), and yaw (z). Each 3D scatter plot was generated using all the environmental conditions available.

TABLE 10. Comparison between the two predictor models. 6DoF refers to sway, surge, heave, roll, pitch, and yaw, and 3 DoF stands for sway, surge, and yaw.

Characteristic	MLP	LSTM
Input variables	6DoF	3 DoF
Input	600 seconds	1000 seconds
Output	100 seconds	400 seconds
Output variables	3 DoF	3 DoF
Training units	55,000 units	10,000 units
Validation units	11,000 units	2,000 units
Trainable parameters	58,872,900	485,227
Training time	short ($\approx 5h$)	long ($\approx 24h$)
Execution time (one prediction)	negligible (real time)	

The scatter plots generated with the MLP prediction are shown in Fig. 24, for RMSE errors; and Figs. 25 and 26 for the Mean and Median errors, respectively. There is a clear separation between the cases with all the lines intact and the cases that have a failure of the mooring line. As the RMSE error depends on the quadratic difference between the measurements, it is impossible to assess the signs of changes when a line fails. This issue no longer occurs with mean and median errors. There is also a clear separation between the four groups of mooring cables of the platform used, indicating that there is a possibility in the future to identify which group the fault belongs to.

The scatter plots generated with the LSTM prediction are shown in Fig. 27, for RMSE errors, and Figs. 28 and 29 for the Mean and Median errors, respectively. It can be seen that here, too, there is a clear separation between the cases with all the lines intact and the cases that have a failure of the mooring line. In the mean and median errors, there is a clear separation between the four groups of mooring lines of the platform used, indicating that there is a possibility in the future of identifying which group the fault line belongs to.

VII. CONCLUSION AND FUTURE WORK

The work aimed to develop a mooring line failure detection system, hypothesizing that the change in platform motions can be a good indicator of when mooring line failure occurs. It was assumed that the platform motion changes in its intensity and frequency after a line failure occurs. Since both ANN models were only trained to respond well to platform motions with all mooring lines intact, this irregular motion can then be seen as a difference between the simulated and predicted platform motions, leading to a significantly higher error score in case of failure. Results of the two ANN models – MLP and LSTM – presented in Section VI correlates with the made hypothesis. The two models were trained and tested on simulated data from Dynasim software.

The Dynasim software was fed real-life environmental weather from Campos Basin of Rio de Janeiro, and the output of the simulation software was the 6DoF motion of the FPSO

(surge, sway, yaw, roll, pitch and heave). These simulator responses were then provided to our networks developed to predict future platform motions using a few previous platform motions. Inputs to our MLP network were 6DoF of simulated data and 3DoF (surge, sway, and yaw) motions for the LSTM network. Outputs of the two models were the surge, sway, and yaw motions of a spread moored platform with 18 mooring lines and 79 risers.

The two models could predict the platform's motions with intact mooring lines, and the error scores between the simulated motions and the model predictions were minimal. For platform motion with a broken mooring line, both models were also able to detect when line breakage occurred by exhibiting a deviation between the predicted and the simulated platform motions, thus confirming breakage has occurred. When compared to errors without line breakage, the error score of this condition reveals an increase in order of magnitude of 10, certifying breakage has occurred.

Both models were then tested on a mild and stormy sea states. For platform motion with mild sea state, both models predicted the oscillations of the 3DoF of the platform, while for platform motion in stormy sea state, both models were unable to forecast the platform's motions. Table 10 shows a comparison between the two trained predictors.

It can be seen that the LSTM model is able to handle a larger input (i.e., more time steps at the input) than the MLP network, and it was also able to forecast a longer output. The forecast time for the LSTM network (400 seconds) was also four times larger than the 100 second forecast time for the MLP network. Therefore, the LSTM model can be used for longer platform motion forecasts than the MLP model. The LSTM network also used fewer platform motion variables as inputs (3 horizontal platform motion variables) than the MLP network, which used surge, sway, yaw, roll, pitch and heave as input. However, they both predicted the same three platform motion features (surge, sway and yaw). As the LSTM training process is slower than the MLP, the number of training units was significantly lower than the MLP training units without affecting the performance of the LSTM in the execution phase. The LSTM model performed better in all cases when compared to the MLP model. Both models are equally quick to predict once they are trained. The differences between predicted and simulated motions were measured in three different errors, RMSE, mean and median errors. They showed a clear separation between cases with a broken line and those with intact lines. Also, we found that the mean and median errors still allow a multiclass classification of failure in the mooring line, enabling the identification of the group of lines to which the broken line belongs. In conclusion, mooring line failure detection using platform motion is viable and finally, the LSTM model was better than the MLP in all test scenarios.

For future work, the implemented predictors will be improved, training them on different drafts and on more platform motions with rough sea conditions to improve the accuracy of the network forecast under these conditions. Since our models were trained and tested on simulated data, we will use real-live platform motions to train and test our implemented predictors. It cannot be overlooked that real data will bring many more challenges, which will need to be treated with care.

Furthermore, a classifier with an ML algorithm will be implemented to make a multiclass classification of the mooring lines' conditions and thus be able to indicate in which group of lines the breakage occurred. In this way, more information will be made available to assist in decision making and scheduling the physical inspection of the mooring cables, giving more security to the floating platforms.

REFERENCES

[1] M. G. Brown, T. D. Hall, D. G. Marr, M. English, and R. O. Snell, "Floating production mooring integrity JIP–key findings," in *Proc. Offshore Technol. Conf.*, 2005, pp. 1–12.
[2] K. Cho, B. van Merriënboer, G. Ç, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *CoRR*, vol. abs/1406.1078, pp. 1–15, Jun. 2014.
[3] M. Chung, S. Kim, K. Lee, and D. H. Shin, "Detection of damaged mooring line based on deep neural networks," *Ocean Eng.*, vol. 209, Aug. 2020, Art. no. 107522.
[4] P. Domingos, "A few useful things to know about machine learning," *Commun. ACM*, vol. 55, no. 10, pp. 78–87, Oct. 2012.
[5] G. Dreyfus, *Neural Networks: Methodology and Applications*. Berlin, Germany: Springer-Verlag, 2005.
[6] E. Fontaine, A. Kilner, C. Carra, D. Washington, K. T. Ma, A. Phadke, D. Laskowski, and G. Kusinski, "Industry survey of past failures, preemptive replacements and reported degradations for mooring systems of floating production units," in *Proc. Offshore Technol. Conf.*, 2014, pp. 1–14.
[7] A. Gensler, J. Henze, B. Sick, and N. Raabe, "Deep learning for solar power forecasting—An approach using AutoEncoder and LSTM neural networks," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2016, pp. 002858–002865.
[8] R. B. Gordon, M. G. Brown, and E. M. Allen, "Mooring integrity management: A state-of-the-art review," in *Proc. Offshore Technol. Conf.*, 2014, pp. 1–19.
[9] J. M. Gumley, M. J. Henry, and A. E. Potts, "A novel method for predicting the motion of moored floating bodies," in *Proc. 35th Int. Conf. Ocean, Offshore Arctic Eng. (ASME)*. New York, NY, USA: American Society of Mechanical Engineers Digital Collection, Jun. 2016, Art. no. V003T02A056.
[10] S. S. Haykin, *Neural Networks and Learning Machines*. New York, NY, USA: Prentice-Hall, 2009.
[11] V. Jaiswal and A. Ruskin, "Mooring line failure detection using machine learning," in *Proc. Offshore Technol. Conf.*, 2019, pp. 1–7.
[12] K. Ma, H. Shu, P. Smedley, D. L'Hostis, and A. Duggal, "A historical review on integrity issues of permanent mooring systems," in *Proc. Offshore Technol. Conf.*, 2013, pp. 1–14.
[13] K. Nishimoto, C. H. Fucatu, and I. Q. Masetti, "Dynasim—A time domain simulator of anchored FPSO," *J. Offshore Mech. Arctic Eng.*, vol. 124, no. 4, pp. 203–211, 2002.
[14] T. A. Pederson, T. S. deGier, A. D. Hall, and S. Allan, "Mooring system life extension using subsea inspection technologies," in *Proc. Offshore Technol. Conf.*, 2013, pp. 1–11.
[15] I. Prislin and S. Maroju, "Mooring integrity and machine learning," in *Proc. Offshore Technol. Conf.*, 2017, pp. 1–12.
[16] D. E. Sidarta, J. Kyoung, J. O'Sullivan, and K. F. Lambrakos, "Prediction of offshore platform mooring line tensions using artificial neural network," in *Proc. 36th Int. Conf. Ocean, Offshore Arctic Eng., Offshore Technol. (ASME)*. New York, NY, USA: American Society of Mechanical Engineers Digital Collection, Jun. 2017.
[17] D. E. Sidarta, J. O'Sullivan, and H.-J. Lim, "Damage detection of offshore platform mooring line using artificial neural network," in *Proc. 37th Int. Conf. Ocean, Offshore Arctic Eng., Offshore Technol. (ASME)*. New York, NY, USA: American Society of Mechanical Engineers Digital Collection, Jun. 2018.

[18] F.-X. Siréta and D. Zhang, "Smart mooring monitoring system for line break detection from motion sensors," in *Proc. 13th ISOPE Pacific/Asia Offshore Mech. Symp.* Mountain View, CA, USA: International Society of Offshore and Polar Engineers, 2018, pp. 1–6.
[19] S. Ukani, W. Maurel, and R. Daran, "Mooring lines–integrity management," in *Proc. Offshore Technol. Conf.*, 2012, pp. 1–5.

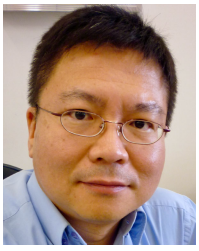
AMIR MUHAMMED SAAD received the bachelor's degree in information system from the American University of Nigeria, Nigeria, in 2013, and the master's degree in computer science from Coventry University, U.K., in 2016. He is currently pursuing the Ph.D. degree in computer engineering with the Escola Politécnica, Universidade de São Paulo (USP), Brazil.

FLORIAN SCHOPP was born in Darmstadt, Germany. He received the B.Eng. degree in electrical and information technology from TU Darmstadt, Germany, in 2018, where he is currently pursuing the master's degree. He joined the Universidade de São Paulo (USP), Brazil, for completing the Double Degree program, in 2019. His research interest includes the development of LSTM neural networks used for identifying mooring line failures in real time.

RODRIGO A. BARREIRA received the degree and the Master of Science degree from Rio de Janeiro Federal University (UFRJ), Brazil, in 2004 and 2006, respectively, where he is currently pursuing the Ph.D. degree with the Department of Ocean Engineering. He is currently a Naval Architect. In 2005, he began in Petrobras as a Naval Engineer. Since then, he has been working with the Research and Development Center, developing research and development projects internally and in partnership with universities. His research interest includes the detection of abrupt change in time series with application in mooring line break detection systems of offshore platforms.

ISMAEL H. F. SANTOS was born in Rio de Janeiro, Brazil, in 1963. He received the bachelor's degree in electrical engineering and the Master of Science degree in applied mathematics from the Federal University of Rio de Janeiro and the Ph.D. degree in computer graphics/virtual reality from the Pontifical Catholic University of Rio de Janeiro. At the Petrobras Research Centre (Cenpes), he has been leading research and development projects using machine learning and data analytics techniques, since 2014. His research interests include the combination of artificial intelligence, computer graphics, and physics-informed machine learning to develop hybrid simulations (data and model driven) in the E&P segment, mainly production, offshore engineering, reservoir simulation, and geophysics.

EDUARDO A. TANNURI was born in Sao Paulo, Brazil, in 1976. He received the degree in mechatronics engineering and the Ph.D. degree in control from the Universidade de São Paulo (USP), Brazil, in 1998 and 2003, respectively. He is currently a Full Professor with the Department of Mechatronics Engineering, USP. He has published 188 papers in conferences and 38 articles in journals. His research interests include non-linear dynamics and control, with applications in autonomous maritime vehicles and ship maneuverability. He has been a member of the International Towing Tank Conference (ITTC) Maneuvering Committee, since 2011. He received the Best Student Paper Award in the IFAC CAMS Conference in 2001 and the National Prizes (the Brazilian Navy Engineering Merit Award in 2017 and the National Agency for Petroleum, Natural Gas and Biofuels Innovation Prize in 2019).



EDSON S. GOMI (Senior Member, IEEE) received the B.Sc. degree in electrical engineering from the Escola Politécnica, Universidade de São Paulo (USP), in 1984, and the Ph.D. degree from The University of Tokyo, in 1996. He is currently an Assistant Professor with the Department of Computer Engineering, Escola Politécnica, USP. He also works on research projects in the areas of oil and gas, glaucoma diagnosis, and law and information technology. His research interests include artificial intelligence and machine learning.

ANNA H. REALI COSTA (Member, IEEE) received the Ph.D. degree from the Universidade de São Paulo (USP), Brazil. She is currently a Full Professor with the Computer Engineering Department, USP. She investigated robot vision as the Research Scientist at the University of Karlsruhe. She was a Guest Researcher with Carnegie Mellon University, working in the integration of learning, planning, and execution in mobile robot teams. She is also the Head of the Intelligent Techniques Laboratory (LTI) and the Director of the Data Science Center, a partnership between USP and Itaú-Unibanco Bank. She is also a member of the Center for Artificial Intelligence (C4AI), USP, in partnership with IBM and FAPESP. Her scientific contributions include artificial intelligence and machine learning, in particular reinforcement learning. Her long-term research objective is to create autonomous, ethical, and robust agents that can learn to interact in complex and dynamic environments, aiming at the well-being of human beings.