A Recurrent Auto-Encoder Neural Network Model for Multi-Channel Ship Pose Estimation to Facilitate Automated Vertical Landings of Aircrafts

Kameron P.C. Palmer, Rishad A. Irani

Multi-Domain Laboratory, Department of Mechanical and Aerospace Engineering, Carleton University, Ottawa, Ontario, K1S 5B6, Canada

April 28, 2022

Abstract

The ability to accurately predict a ships future roll, pitch, and heave motions from measured motions is vital for many maritime applications. Without accurate and reliable predictions autonomous systems may make choices that could cause damage to themselves, equipment on the ship's deck, or harm the ship's crew. Current state-of-the-art methods for predicting ship pose rely on either analytical methods, which make approximations on underlying dynamics of a ship's motion, or on older data driven methods. To facilitate the automated or autonomous vertical landing of an aircraft on the deck of a maritime vessel the ability to accurately predict the ships future roll, pitch, and heave motions is vital. Current state-of-the-art methods for predicting ship pose rely on either analytical methods, which make approximations on underlying dynamics of a ship's motion, or on older data driven methods. The work presented in this paper uses a modern Recurrent Neural Network (RNN) architecture constructed using Gated Recurrent Unit (GRU) cells and an auto-encoder architecture in order to predict a ship's motion. The proposed GRU Autoencoder model is compared against the more common feed forward Neural Network (NN) non-linear autoregressive exogenous (NARX) model. Both NN models are tested for robustness by studying the impact that noise, sea state, and ship model has on the overall performance. It was found that the GRU Autoencoder model outperforms the NN NARX model in almost all scenarios and was more robust. Additionally, guidelines for creating a training dataset that will create more robust prediction models are also presented.

I. Introduction

Ships in the open ocean environment are expected to be able to operate in a wide range of ocean conditions. Ocean waves will induce pitching, rolling, and heaving motions on the ship which will negatively impact on-board operations. The ability to make quick and accurate predictions to the ships motion is important for tasks such as safely operating shipboard cranes [1] or vertical landings on the deck [2]. The roll, pitch, and heave motions can be measured using one or more sensors in order to construct time signals which a signal prediction model can use to estimate future ship motion values.

This work seeks to improve the state-of-the-art for ship motion prediction methods applied for automating Uninhabited Aerial Vehicle (UAV) vertical landings on ship decks. During flight, the UAV tracking phases, flight control and navigation are well studied [2][3]. However, the final descent and landing on moving platforms is still an unsolved problem due to the inherent difficulties of operation in this kind of scario [4]. The work herein examines methods to improve vertical landings on a moving vessel. The use of mounted LIDAR sensors allows for the control system of the UAV to be independent of the ship it is landing from, removing hardware requirements from vessels and allowing for use on multiple ships. The UAV's landing system would rely solely on its own sensors and signal prediction model to perform a safe landing. The goal of the current work is to predict the roll, pitch, and heave of a ship to facilitate a safe landing of a UAV [2][3]. Broadly speaking, ship motion is a highly coupled stochastic non-linear system; the six-degrees of freedom of the ship motion have non-trivial dependencies on the stochastic ocean waves dynamics, the ship hull structure, and the relative orientation to the sea [5].

Analytical prediction models use approximations of the underlying description of the ships motion. By making an approximation, an analytical model may be applied to a wide variety of cases at the potential cost of performance in a specific scenario of interest. Fast Fourier Transform (FFT) methods are a typical analytical method that aim to estimate ship motion as the sum of sine waves and have been used to predict various ship motions [2][3][6][7]. Autoregressive (AR) models are a stochastic method that use recent measured history to predict a step into the future. AR models have been successfully implemented for predicting ship roll, pitch, and heave motion [8]. Other methods such as Prony analysis [9] and variant ellipsoid methods [4] have also been applied to predicting ship motion.

While the models listed above have all found success there is potential for improvement in data driven models which use measured or simulated data in order to create a prediction model tailored to a specific problem. Montáns et al. [10] published a review of data driven approaches in a wide range of engineering applications and noted that computational power and data availability area increasing and will create opportunities for new approaches to existing problems. Neural Networks (NN) are a form of data driven prediction models which have been applied to ship motion prediction and are becoming increasingly common. NNs have been used to predict ship and roll motions [11][12][13] as well the ship heave motion [14][15].

Feed forward NNs are adept at performing predictive tasks, but were mainly developed in the 1980s and there has been significant advances to NNs since. Recurrent Neural Networks (RNNs) [16][17] are an adaptation of NNs that use recurring connections to the same sets of weights to understand and make use of ordered data such as time signals, text, speech, and music. Both the Elmann and Jordan RNN architectures have been applied to predicting ship motion [18][19]. The Elmann RNN has been shown to outperform analytical models constructed with Kalman filters [19] and, when provided with sufficient datasets, they also outperform the feed forward NNs [20][21][22].

A historical draw back with the Jordan and Elmann RNNs was a vanishing gradient problem that would prevent training RNNs using longer sequences of data [23]. This problem was address by Hochreiter et al. with the creation of the Long Short-Term Memory (LSTM) RNN cell structure [24]. By introducing two internal structures, referred to as gates, to the RNN cell that is responsible for determine what information from previous timesteps is significant, the LSTM does not suffer from the vanishing gradient problem, allowing for longer sequences to be used. Recently, the LSTM has been applied successfully for predicting ship motion [25].

The Gated Recurrent Unit (GRU) is an adaptation of the LSTM that has less weights that need to be trained when compared to the LSTM, making it easier to train and quicker to calculate [26]. It has been shown that GRU models outperform the Jordan RNN and perform on par with the LSTM [27]. The GRU models have been applied to predicting ship roll, pitch, and heave motions; where they were found to be comparable to the LSTM, though the GRU model was quicker to train [28].

Further development on the RNN structure was done by dividing the RNN into two substructures that each resemble a full RNN [29]. The first substructure, named the encoder, takes the input signal and compacted the information into only the significant features that describe it, as determined by training. The second substructure, named the decoder, takes the results from the encoder and decompressed it to create the prediction. The combination of the encoder and decoder substructures, referred to as an autoencoder, improved English to French when compared to other machine translation methods. Additionally, the autoencoder could handle long sequences well and has found success at predicting vehicle trajectories [30]. The GRU autoencoder structure utilizes the strengths of the GRU RNN for sequence learning and prediction as well as the advantages of the divided autoencoder structure, and thus can be used to create a model which improves on the state-of-the-art for ship motion prediction.

To the authors of this works best knowledge, the autoencoder RNN architecture has not been applied to the task of predicting ship motion. Additionally, most applications use NARX structures which instead of predicting full, multi-channeled signals for future ship motions. Data has also been limited when creating NN based models, with most datasets coming from a single set of measurements. Prior work has commented that the impact of environmental factors, such as wave height and wind speed, should be considered when constructing NN based predictive models [13]. However, limit testing for NN models has also been minimal, with few authors including the impact of noise, varying sea state, and ship model in their analysis.

The work presented in this paper seeks to improve and contribute to the state-of-the-art by applying an auto-encoder RNN structure with GRU cells to make multivariate predictions of future roll, pitch, and heave motion based on recently measured data. The models presented in this work are constructed to only make use of instantaneous roll, pitch, and heave measurements that would reflect those obtained from UAV mounted LIDAR sensors so that the models can be later implemented into an automatic landing system. The data sets used in this work are generated from a combination of 21 different simulations. A comparison with a single step-ahead NN NARX model, which is more common in literature, is performed to demonstrate the improved performance of the GRU autoencoder [11][12][13][14][15][18][28]. Lastly, the impacts and measurement noise, varying sea state, and changing ship models are considered in order to limit test the models and evaluate their ability to generalize performance under a range of application focused extremes with the intent of creating a single, once-trained NN model capable of handling a wide range of scenarios.

The remainder of this paper is as follows: Section II introduces the construction of the datasets used in this work and represent the "problem" to be solved while Section III presents the formulation of three prediction models used to examine the problem and highlight their baseline performance. Section IV will show studies of the impact of noise, varying sea states, and changing ship models.

II. Data Construction: ShipMo3D Simulations

Neural Network models must be trained against sets of data which should be large and varied enough in order to fully describe the application scenario the model will perform in. If the training data is limited in some way, such as by being small or having data that are too similar, the performance of the NN will reflect the limited data and not the generalize well. Furthermore, attempting to use a NN model with data from a set unlike the one it was trained on is expected to negatively impact performance, potentially in ways with effects that cannot be easily predicted. Small data sets have been a limiting factor in developing NN predictive models for predicting ship motion. The work presented in this paper uses simulations from the validated ship motion modelling software ShipMo3D [5][31] to construct the training and validation datasets. Within the current study, LIDAR or similar measurements are not used as an input to the proposed system. Previous work has developed a method to determine the world frame ship motion when using a UAV mounted LIDAR system [3][2]. Thus, the current work uses the world frame ship motion simulations to assess the proposed system while the case studies, section IV, provide insight on external factors which impact measurement.

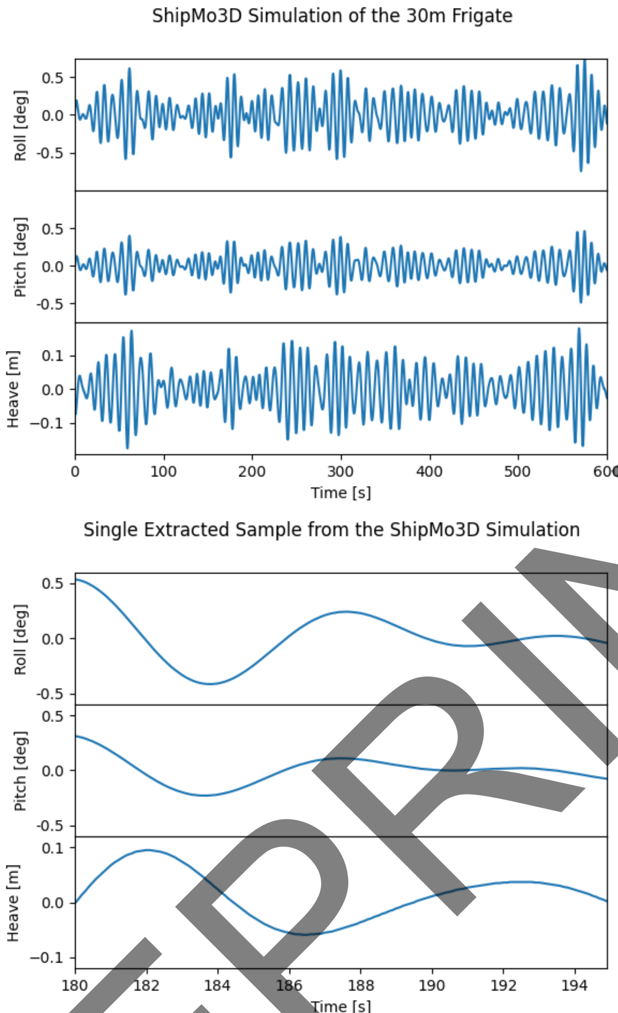

Fig. 1. **[top]** The roll (top row), pitch (middle row), and heave (bottom row) ship motions sampled from the ShipMo3D simulation where the ship was travelling at 10 kt with an angle of attack of 120 deg. **[bottom]** A typical 15 sample that is extracted from the simulation that would be used as a single input for a prediction model.

Each simulation use a 30 metre vessel and sea state 2 ocean conditions, ship speeds of 6 kn, 8 kn, and 10 kn and headings of 0, 30, 60, 90, 120, 150, and 180 degrees were used to create 21 simulations. The various combinations of ship speeds and headings provide a better representation of the general relationship of waves on the ship. Each simulation is ran for a total of 6 minutes and is sampled at 10 Hz.

Fig. 1(top) shows a full ShipMo3D simulation where the ship was travelling at 10 kt with a heading of 120 deg. Fig. 1(bottom) shows a 15 s sample that would be use to construct a typical input for the prediction models. While ShipMo3D provided the full kinematics of the ship, only the pitch, roll, and heave motions are extracted for use in this work. Additionally, while this work is limited to roll, pitch, and heave measurements as the input and target signals any measurable signals, such as the rates and accelerations, may be used as inputs or targets. The models presented in this work will require that all input channels share the same sampling frequency and that all target channels share the same sampling frequency.

The individual channels are normalized separately in order to have zero mean and a variation of 1. The normalization is done for two reasons, firstly, normalizing features inputted into a Neural Network increases the numeric stability of training and improves the speed of convergence towards a minimum, and secondly, without the normalization the channels may take values on different scales. Normalizing the data so that each channel is on the same scale prevents errors in any one channel from dominating the others and adding bias to the training process. The data signals are normalized using

$$\hat{x} = \frac{x - \mu}{\sqrt{\sigma^2}}, \tag{1}$$

where x is original data point, μ and σ^2 are the mean and variance calculated across all data points in all simulations, and $\hat{x}$ is the new normalized data point. The mean values, across all 21 simulations, are near zero for roll, pitch, and heave. Prior to normalization the roll variance is $\sigma^2_{roll} = 0.624$ deg^2, the pitch variance is $\sigma^2_{pitch} = 0.0785$ deg^2, and the heave variance is $\sigma^2_{heave} = 4.784 \times 10^{-3}$ m^2 deg^2. After normalization the simulation data will have a mean of zero and a variance of one when constructing the individual data sample pairs for training and testing.

The input signal duration is chosen to be 15 s in order to ensure the measurements contain at least one peak wave period for up to sea state six [32], which is the highest sea state a UAV may attempt a vertical landing. The target signal duration is chosen to be 5 s, which is a typical descent for the UAV that allows for time to abort dangerous landings [2]. In most cases a NNs which is trained on more data will outperform NNs trained on less. In order to create the largest dataset possible, every continuous 20 s interval is extracted from the 21 simulations and used to construct input and target data pairs. In total, 121 800 input-target data pairs are created. The data pairs are shuffled randomly and split into a training data set consisting of 80% of the total data, 97 440 data pairs, and a testing data set consisting of the remaining 20% of the total data, 24 360 data pairs. If the data were not shuffled a sampling bias towards a single simulation could negatively impact the batch gradient descent training process, leading to an under performing predictive model.

The training data set is used to optimize the NN weights against a Mean Squared Error (MSE) loss function and validation data set is used to evaluate performance. Using the validation data set a comparison can be made between prior NN NARX based models and the proposed GRU autoencoder model.

III. Ship Motion Prediction Models

This section presents the structure of the modern RNN architecture, constructed for making multi-channel sequence-to-sequence predictions using an autoencoder structure and Gated Recurrent Unit (GRU) cell structure. The proposed GRU Autoencoder model is also compared to a more common feed forward Neural Network (NN) non-linear autoregressive exogenous (NARX) model. All models were trained and tested using the Tensorflow [33] framework; however, any similar NN software toolkit would work.

As the data used has a zero mean the NNs presented in this work do no include bias elements. While it is possible to include a bias in NN and allow training to conclude that the bias element must be zero in order have this property, it would be inefficient when compared to applying prior knowledge and forcing the bias to be zero.

A. Neural Network NARX model

Feed-forward Neural Networks (NNs) are data driven models that are capable of learning important features from a set of data in order to make accurate predictions. NNs can be used to construct nonlinear autoregressive exogenous (NARX) models which are commonly used in ship motion prediction models. Different NN NARX models can be constructed by varying hyperparameters such as the number of steps ahead, input values, and target values. Figure 2 shows the NN NARX model used in this work; the NN NARX uses roll $r(t)$, pitch $(p(t)$, and heave $h(t)$ motions, represented by the combined value $x(t)$ and flattened into a single vector, as the inputs. The NN has two sets of weights, one W_1 in a dense hidden layer with a rectified linear unit (ReLU) activation function and a second W_2 in the dense output layer. The total number of trainable parameters in the network is equal to sum of the sizes of W_1 and W_2. As the input and output vector sizes are determined by the data the only way to control the total number of trainable parameters is to adjust the number of hidden neurons, which is the columns and row size if W_1 and W_2 respectively.

Only the measured values from the timesteps between t_0, the starting time of the sampling window, and t_s, the length of the sampling window are included in the input vector. The final output is a vector containing the roll, pitch, and heave motions of the ship at the next timestep. The oldest measurements in the input vector are removed and the new prediction is appended to the input vector in order to construct the next input. Iterating the prediction, removal, and appending process creates the NN NARX model which can be used to construct indefinitely long predictions.

The NN NARX model is trained using batch gradient descent with a batch size of 32 and a learning rate of 1×10^{-4}. The chosen loss function is the Mean Squared Error (MSE) of normalized roll, pitch, and heave motions. Training is considered sufficient when the change in training loss is less than 1×10^{-4}. The NN NARX model trains quickly as it only computes 3 values. However, since it is only predicting the ship motion at a single time step it may propagate errors when constructing long term predictions. The architecture of the NN NARX model does not consider the order of the inputs and therefore it may not be the ideal architecture for handling time dependent signals, such as ship motion. While the NN NARX model has been successful at predicting ship motion [11][12][13][14][15][18][28] the GRU Autoencoder model, which is designed to consider ordered inputs such as time signals, is expected to improve performance. Furthermore, the GRU Autoencoder trains against its

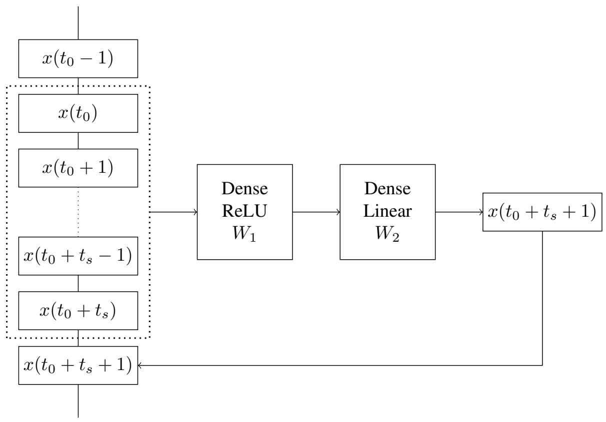

Fig. 2. A visual that describes the information flow in the NN NARX model. The inputs $x(t)$ consist of the roll $r(t)$, pitch $p(t)$, and heave $h(t)$ motions. Each prediction is made using information from the sampling time between t_0 and $t_0 + t_s$. After each prediction the oldest measurements are removed from the input vector and the predictions are appended to the input vector. Iterating the prediction and cycling allows for long term predictions using the single model.

ability to make full signal predictions which naturally reduces error from the predictions that the NN NARX would introduce. It is believed that the GRU Autoencoder model will outperform the NN NARX model by a significant amount.

B. GRU Autoencoder

Recurrent Neural Networks (RNNs) are a class of Neural Networks that are structured to considered the order of inputs. RNNs are built from one or more cell structures that are provided inputs in a sequential manner and produce outputs at each step. They are meant to handle sequences of information, such as the ship motion time signals. Unlike the NN NARX model in Sec. III-A an RNN can be structured to create sequence outputs, such as full time signal predictions of ship motion.

RNN cells take up to two inputs, the current time signal and the prior cell state, and provides up to two outputs, the current prediction and the current cell state, for each time step. At each time step the information being passed forward in time, referred to as the cell state, will be updated using the previous cell state. Figure 3 shows the autoencoder RNN model, which segments the RNN into two components. The first component is the encoder, which will take in the signal inputs x_t and cell states at each time step, pass the cell states forward, but not return any signal. The second component is the decoder, where the cells will only take prior cell states as their inputs, pass the cell states forward, and return a signal at the current time step. The encoder and decoder use separate sets of weights and allow training to let the data determine which components of the inputs are most important and how those components form the predictions x_{t+n}. The cell states are used to pass information forward through time and are marked by arrows in Figure 3

Each of the cells shown in Fig. 3 represent a recurrent unit cell which defines the type of RNN that is being used. The older Elmann and Jordan RNNs created cell states that would directly pass information forward with each time step. However, the GRU cell modifies the cell state before passing the information forward. The GRU cell structure shown in Fig. 4 is made with

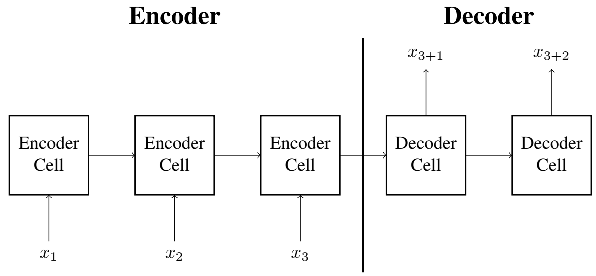

Fig. 3. The structure of an Autoencoder RNN model that uses three timesteps in its input sequence and calculates two timesteps in its output sequence. The Encoder is responsible only for determining what is important from the inputs x_t and does not produces predictions at any timestep. The Decoder takes the important information from the encoder and uses it to produce the outputs x_{t+n} and does not consider any input information directly.

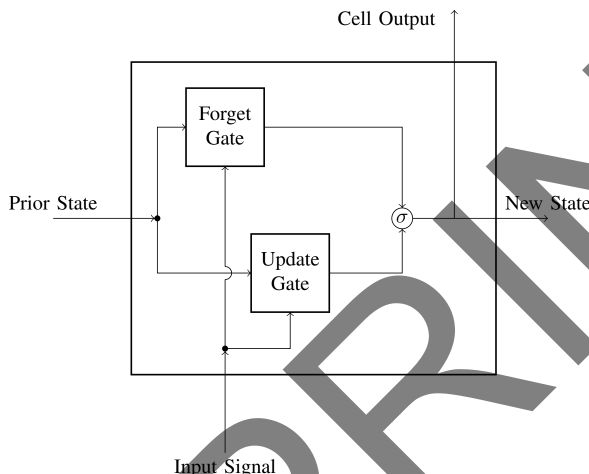

Fig. 4. A high-level visualization of how a GRU cell is structured. Both the current input and prior state are used in both the Forget and Update gates. Each gate has its own weights and will produce independent values which are used as inputs for a sub-function, marked by σ. The result is then passed forward as a new state and as the cell output for that timestep. Each of the Encoder and Decoder boxes in Fig. 3 are this cell structure.

two gate structures that each contain a set of trainable weights. The forget gate determines what information from the prior cell state is significant at the current time step and suppresses the information that is not relevant. The update gate determines what information from the current input should be emphasized in order to calculate the new cell state. The outputs of both gates are combined together and passed through a sigmoid activation function in order to create the new cell state.

The GRU Autoencoder was trained using nearly the same methodology as the NN NARX model was in Sec. III-A. While the NN NARX model was trained for its ability to predict only the most immediate timestep, consisting of 3 values for roll, pitch, and heave, the GRU Autoencoder model was trained for its ability to predict full signals for each motion over a period of 5s at 10 Hz for a total of 150 values. The GRU Autoencoder model will almost certainly contain more trainable weights than the NN NARX model and as a consequence, will take longer to train; however, once trained it will perform significantly better.

C. Model Comparison

The performance of each of the two models described above were compared against each other by the Mean Squared Error (MSE). As described in Sec. II the three signal channels are normalized separately and are dimensionless. The MSE value is therefore also dimensionless and is calculated as the MSE of all channels. Since the objective is to make full signal predictions the MSE is calculated over all timesteps in the target signals. The GRU Autoencoder model inherently makes full signal predictions and the NN NARX model is iterated until the full output signal has been constructed.

As data driven models, the NN NARX model and the GRU Autoencoder model has a large number of parameters to select in order to optimize predictions against the training dataset. Both the NN architecture and the number of parameters will impact the performance of the models. A set of 30 sizes of weights were tested for the NN NARX and 15 layer sizes were tested for the GRU Autoencoder. All other hyperparameters are held constant as their impact on performance would be minimal when compared to the difference between the performance of the two architectures and the primary focus of the current study is to evaluate the advantages of the modern GRU Autoencoder NN structure. The MSE on the testing dataset is show in Fig. 5; the red circles show the testing MSE for the NN NARX models and the blue squares shows the testing MSE for the GRU Autoencoder models.

For the NN NARX there is an upwards trend in the testing MSE as the number of trainable parameters increases, indicating that the NN NARX model has a tendency to over-fit its parameters. The best performing NN NARX model had a layer size of 16, corresponding to 7248 trainable parameters, and a testing MSE value of 0.3747. The worst performing NN NARX model had a layer size of 232, corresponding to 105 096 trainable parameters, and a testing MSE value of 2.687.

The GRU autoencoder models did not suffer from over-fitting as the NN NARX models did and showed a slight decreasing trend in testing MSE values as the number of trainable parameters increases. The best performing GRU Autoencoder model had a layer size of 120, corresponding to 88 920 trainable weights, and a testing MSE value of 0.0495. The worst performing GRU Autoencoder model had a layer size of 24, corresponding to 3960 trainable parameters, and a testing MSE value of 0.1515.

All of the GRU Autoencoder models outperformed all of the NN NARX models, which is attributed to the data driven nature of the GRU Autoencoder model allowing to learn the ship dynamics from the data directly, the architecture of the model which considers time ordered inputs, and being trained against making full time signals instead of single timesteps. The proposed GRU Autoencoder is arguably the better performing model for predicting ship motion.

For the remainder of the work presented in this paper the best performing NN NARX model with layer size of 16 and a corresponding 7248 trainable parameters is chosen to evaluate performance. The GRU Autoencoder model with layer size

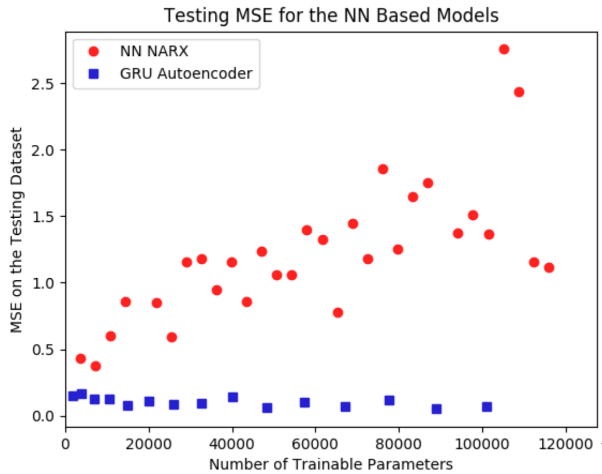

Fig. 5. The testing MSE values for various weight sizes sampled for the NN NARX model, red circles, and the GRU Autoencoder, blue squares. Every GRU Autoencoder models trained out performed every NN NARX model trained regardless of the number of trainable weights.

of 32, corresponding to 6816 trainable parameters and a testing MSE of 0.1243 is used to evaluate the GRU Autoencoder models. The increase in performance by adjusting the number of parameters in the GRU Autoencoder is notably less than the difference in performance between the GRU Autoencoder models and NN NARX models. In order to examine the performance differences between the two models is due to the choice of architecture, the GRU Autoencoder model with the closest number of trainable parameters to the best NN NARX model is chosen to represent the GRU Autoencoder models.

A sample prediction of the NN models is shown in Fig. 6. The solid black lines mark the input signals that the predictive models are provided and the solid green lines are the target signal that the models are attempting to reproduce. The red dashed line is the prediction made by the NN NARX model and the blue dash-dotted line is the prediction made by the GRU Autoencoder. Both the NN NARX and GRU Autoencoder models are able to make good predictions on the roll channel with the GRU Autoencoder predicting most of the channel to high accuracy and the NN NARX only losing accuracy after the second local minimum. The pitch and heave channels show the advantage of the GRU Autoencoder model over the NN NARX model. In both channels the NN NARX makes a large deviation from the target signal while the GRU Autoencoder remains much closer. As the GRU Autoencoder model is trained to value all time steps equally it is common for minor continuity issues to appear in the first few timesteps of the prediction, as seen in the pitch signal. While the GRU Autoencoder corrects the error from the discontinuity it must be considered for applications using real-time predictions.

The data used for training and testing purposes will likely not reflect the data observed in application. Varying factors such as noise in the signal measurement, changing sea states, and different ship models must be expected. As changing the nature of the data will impact the data driven models it is important to understand how applying the GRU Autoencoder model different situations affects performance.

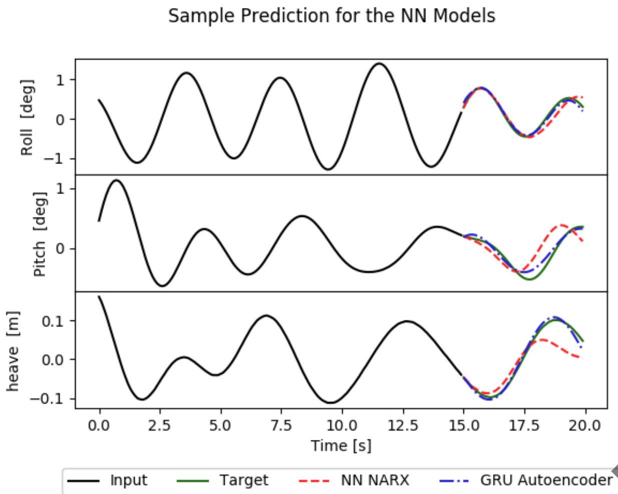

Fig. 6. A typical sample prediction for the NN NARX, the red dashed line, and the GRU Autoencoder, the blue dash-dotted line. The input signal provided to the models is shown as the solid black line and the models are attempting to predict the future motion shown by the solid green line.

IV. Case Studies

The data measured in application scenarios is rarely identical to the data acquired *a priori*. Measurement noise, varying sea state levels, and different ship model dynamics can be expected and should be accounted for when implementing a ship motion prediction routine. As the NN NARX and GRU Autoencoder models require training against prior data to function it is critical to understand the impact of how these models react when presented with different data. While transfer learning methods may be applied to account for these differences the process of retraining during application can be computationally expensive and time consuming. This section presents a study of how noise, sea state, and ship model impact the performance of the NN NARX and GRU Autoencoder models and aims to assist in constructing guidelines for creating a model which can generalize a wide range of scenarios natively.

A. The Impact of Noise

The data described in Section II does not contain noise and so may not reflect in-situ data well as sensor data is rarely noise free. Thus, we seek to understand how the NN models presented handle noise similar to real sensor data. Noise is added to the simulated sea state 2 30 m vessel input signals by applying random, normally distributed values with zero mean and a variance of σ^2. The noise levels are set to be equal across each of the normalized motion channels and cover a range up to $\sigma^2 = 1.0$, which correspond to pitch, roll, and heave variances of 0.079 deg ^{2}, 0.624 deg^2, and 4.78×10^{-3} cm^2 respectively.

Figure 7 shows how the NN models perform as the noise level increases, as judged by the MSE value on the full testing dataset. The solid blue line shows the best performing NN NARX model and the dash-dotted red line shows the selected GRU Autoencoder model.

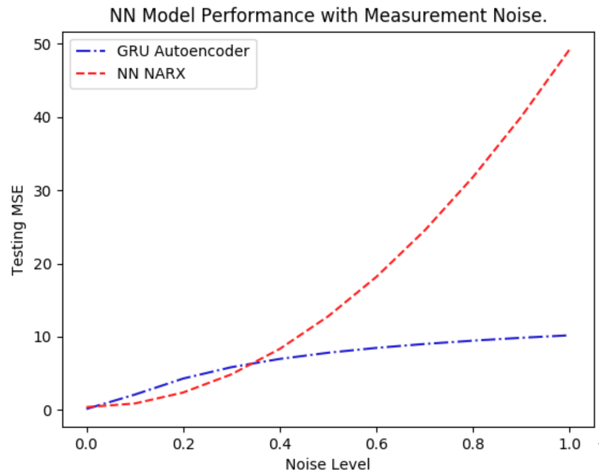

Fig. 7. The MSE values for the NN NARX model, shown by the dashed red line, and the GRU Autoencoder model, shown by the blue dash-dotted line, for various levels of noise in the testing inputs. As both models were trained without noise neither model is able to handle noise well.

Figure 7 shows how the testing MSE value varies when they modes are presented with noisy inputs; noting that the models in the figure are trained with noise free data. The NN NARX model, red dashed, and the GRU Autoencoder, blue dash-dotted, both have increasing testing MSE values as the noise level increases. For low levels of noise, below 0.35, the NN NARX has a lower MSE when compared to the propsed GRU Autoencoder model. The advantage of the NN NARX is likely due to the iterative calculation being able to correct well for small levels of noise. Unlike the NN NARX, whose MSE value increases exponentially as the noise increases beyond 0.35, the GRU Autoencoder MSE increases logarithmiclally. The GRU Autoencoders better performance on inputs with higher noise is due to training against full signals instead of individual timesteps, which allows the GRU Autoencoder to extract the underlying motion in the measured noise while making predictions.

For roll, pitch, and heave Fig.8 and 9 show typical sample for how the NN models perform when presented with a lower noise level of 0.2 and a higher noise level of 1.0 respectively. The true input signal is marked by the solid black line and the measured signal, which contains noise, is marked by the solid grey line. As seen in Fig. 8 the presence of noise has a dramatic affect on the performance of both NN models. In each of the roll, pitch, and heave motions the NN NARX model propagated the noise into its predictions. The GRU Autoencoder did not show noise like behaviour in its predictions, but as shown in Fig. 7, the quality of predictions is lower than the NN NARX. With the higher noise level in Fig. 9 the NN NARX has difficulties preventing itself from propagating errors, which are noted by the large overshoots in each of the motion channels. In comparison, the GRU Autoencoder in Fig. 9 stays close to the target motions, although with unsatisfactory prediction quality.

As NN models are data driven their prediction capabilities will be dependant on how much the training dataset matches the data measured in application. By including noise in the training data a NN model can learn to separate the underlying motion when making predictions. Fig. 10 and Fig. 11 plots the testing MSE of the NN models after being trained on data with a

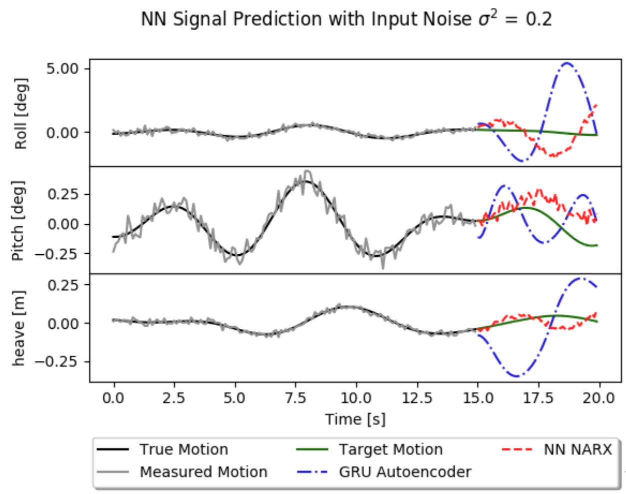

Fig. 8. A typical sample of the prediction made from the NN models, trained without noise, on inputs containing noise with a normalized variance of 0.2, which is a noise level where the NN NARX is expected to outperform the GRU Autoencoder, as shown in Fig. 7.

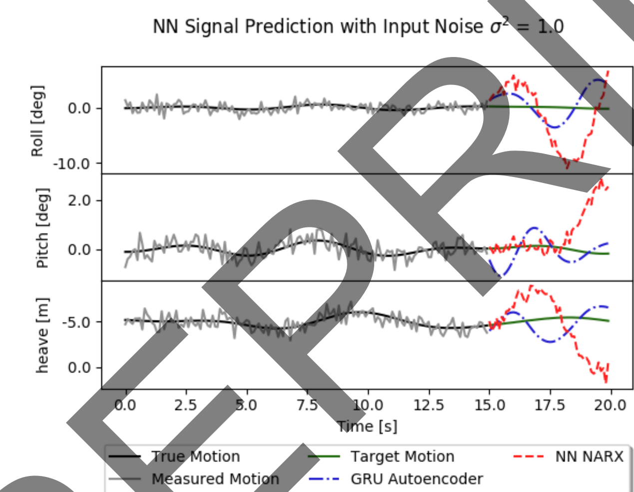

Fig. 9. A typical sample of the prediction made from the NN models, trained without noise, on inputs containing noise with a normalized variance of 1. At higher noise levels, such as shown here, the GRU Autoencoder is expected to outperform the NN NARX, as shown in Fig. 7.

noise level of $\sigma^2 = 0.2$ and a high noise level of $\sigma^2 = 1.0$ respectively. The red dashed line marks the performance of the NN NARX, the blue dash-dotted line marked the performance of the GRU Autoencoder. The characteristic of exponentially increasing testing MSE value for the NN NARX is visible and logarithmically increasing testing MSE for the GRU Autoencoder can be seen. The results of Fig. 10 and Fig. 11 the NN NARX does not outperform the GRU Autoencoder regardless of the noise level, indicating that the GRU Autoencoder architecture and training is better able to extract the underlying motions.

Table I shows the testing MSE values for the NN models when trained and tested on various noise levels. The NN NARX and GRU Autoencoder models performed better when trained with noise than when trained without. When trained without noise and presented with data that contained no noise the NN NARX and GRU Autoencoder models had MSE values of 0.3774 and 0.1242, respectively. When the training data had a noise level of $\sigma^2 = 0.2$ and the testing data contained no noise,

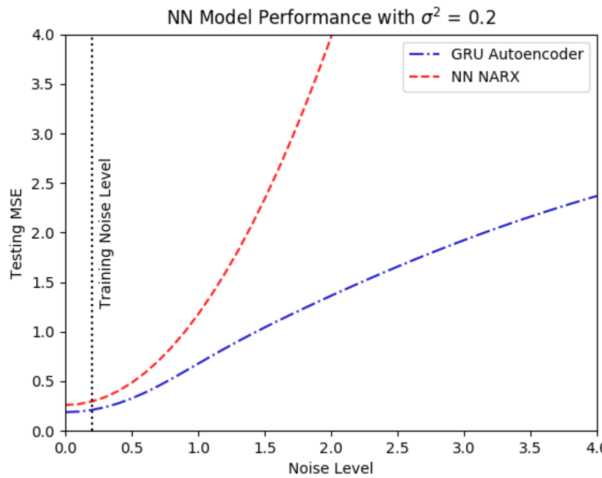

Fig. 10. The testing MSE values for the NN models after being retrained with data containing noise with normalized variance of 0.2. The NN NARX model, shown by blue cross, keeps its exponentially increasing trend. The GRU Autoencoder model, shown by red crosses, keeps its logarithmically increasing trend, though leveling off at a much higher noise level than when not trained with noise. Both NN models perform better that when not trained with noise and the GRU Autoencoder out performs the NN NARX at all noise levels.

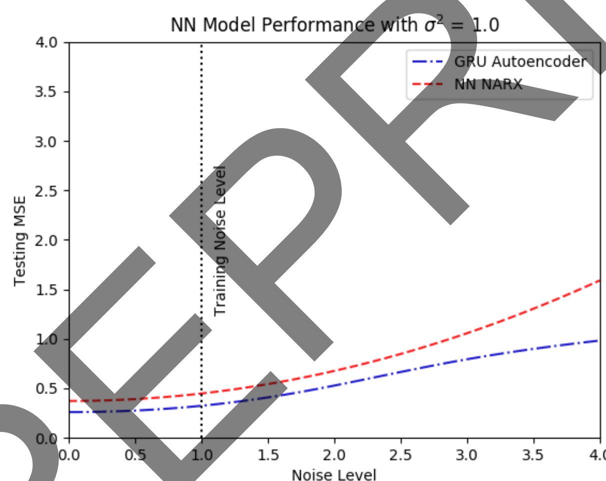

Fig. 11. The testing MSE values for the NN models after being retrained with data containing noise with normalized variance of 1.0. The NN NARX model, shown by blue cross, keeps its exponentially increasing trend. The GRU Autoencoder model, shown by red crosses, keeps its logarithmically increasing trend, though leveling off at a significantly higher noise level than when not trained with noise. Both NN models perform better that when trained with little to no noise and the GRU Autoencoder out performs the NN NARX at all noise levels by a wide margin.

the NN NARX and GRU Autoencoder models had testing MSE values of 0.2612 and 0.1885, respectively. The noise trained NN NARX demonstrated an improvement, even when not evaluating noisy inputs. In comparison the GRU Autoencoder had a decrease in performance.

When trained with data containing a noise level of $\sigma^2 = 1.0$ and tested on data with the same level of noise, the NN NARX and GRU Autoencoder had MSE values of 0.4483 and 0.3243 respectively. Compared to the models that were not trained on noise, the improvement in performance is two orders of magnitude.

When presented with testing data that contained no noise the MSE values for the NN NARX and GRU Autoencoder that were trained on data with a noise level of $\sigma^2 = 1.0$ were 0.3729 and 0.2606 respectively. When presented with testing data

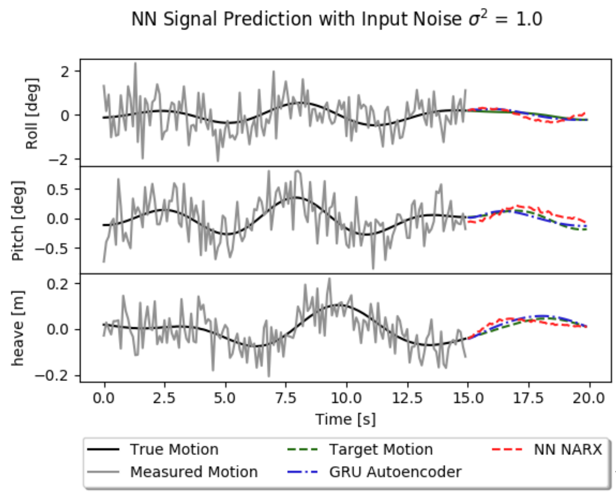

Fig. 12. A typical sample of the prediction made from the NN models, trained with noise with a normalized variance of 1.0 on inputs containing noise at the same level. Both the NN NARX models are able to extract the underlying motions and make predictions despite the high level of noise present in the inputs.

TABLE I
THE TESTING MSE VALUES FOR THE NN NARX AND GRU AUTOENCODER MODELS WHEN TRAINED AND TESTED ON VARIOUS LEVELS OF NOISE. TRAINING THE MODELS WITH NOISE LOWERS PERFORMANCE ON TESTING SAMPLES WITHOUT NOISE BUT GREATLY INCREASES PERFORMANCE ON TESTING SAMPLES WITH NOISE.

		Testing Noise Level σ^2		
Model	Training Noise Level σ^2	0	0.2	1.0
GRU Autoencoder	0	0.1242	4.274	10.19
	0.2	0.1885	0.2109	0.6776
	1.0	0.2606	0.2629	0.3243
NN NARX	0	0.3774	2.367	49.28
	0.2	0.2612	0.2960	1.176
	1.0	0.3729	0.3759	0.4483

that contained a noise level of $\sigma^2 = 0.2$ the high noise trained NN NARX and GRU Autoencoder models had testing MSE values of 0.3759 and 0.2629, respectively. When presented with testing data that contained a noise level of $\sigma^2 = 1.0$ the high noise trained NN NARX and GRU Autoencoder models had testing MSE values of 0.4483 and 0.3243, respectively.

The same sample from Fig. 9 is shown again in Fig. 12, but with predictions made from NN models that have been trained on the input noise level of $\sigma^2 = 1.0$ and the NN models perform significantly better when trained with noise when compared to when they are trained without noise, as was the case in Fig. 9.

The NN model behaviour when using datasets that contain noise indicate that NN based motion prediction models should include noise in training. Furthermore, the noise included should be at the same level, or above, what would be encountered in application. When trained with noise the GRU Autoencoder model outperformed the more commonly used NN NARX model, showing a clear advantage of the proposed prediction model. For the current application, the results show that the common practice of pre-filtering the input signals is not necessary as the NN models are able account for the noise in the data if properly trained.

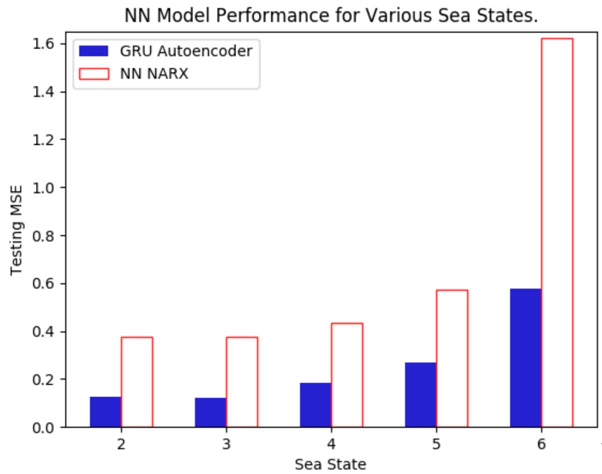

Fig. 13. The NN models are both trained on sea state 2 data and the testing MSE for sea states 2 through 6 are shown. The GRU Autoencoder model, shown as solid blue in the left column, out performs the NN NARX, shown as hollow red bars in the right hand column, for all sea states. As expected, the more complicated sea state 6 waves are the hardest to predict for both NN models.

B. The Impact of Sea State

The NN models presented in Sec. III were trained and tested using data from sea state 2 simulations. In practical applications various sea states will be experienced by a system. Figure 13 shows the testing MSE values for sea states 2 through 6 for the two NN models. As suggested by Fig. 13, the GRU Autoencoder model outperforms the NN NARX model in all sea states. At sea state 2 the NN NARX had a testing MSE of 0.3774 and the GRU Autoencoder had a testing MSE of 0.1242. At sea state 6 the NN NARX had a testing MSE of 1.620 and the GRU Autoencoder had a testing MSE of 0.5774.

The overall impact of increasing sea state was not as significant as introducing noise. Increasing the sea state makes the signal inputs more complicated by changing both the range of possible amplitudes and adding additional underlying modes. The reason that increasing sea state did not impact the models performance as much as adding noise is due to of the normalization of Eq. (1). The normalization brings range of possible values in higher sea states to within a specified, dimensionless range, that is similar among all sea states. By normalizing the data the only increase in signal complexity comes from the increase in underlying modes.

From Fig. 13 when the NN models are trained on only the simpler sea state 2 data they are not able to handle the more complicated sea state 6 data. The NN models can be set up to handle multiple sea states simultaneously by including multiple sea states in the training data. Fig. 14 shows the NN NARX, in red hollow bars, and GRU Autoencoder, inblue solid bars, models after they were retrained to include all data from sea states 2 through 6. The NN NARX model had an average testing MSE value of 1.136, with the highest value of 1.315 occurring for the sea state 6 data and the lowest value of 1.052 occurring for the sea state 4 data. The GRU Autoencoder model had a significantly lower average testing MSE of 0.1306, with the

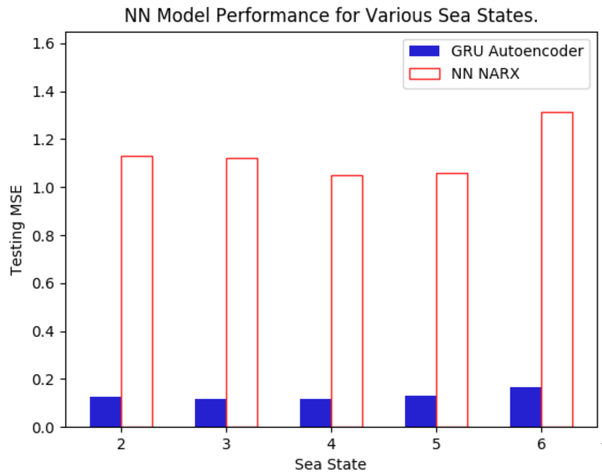

Fig. 14. The NN models after being retrained on data that contains equal portions from each sea state 2 through 6. The GRU Autoencoder model, shown as solid blue in the left column, out performs the NN NARX, shown as hollow red bars in the right hand column, for all sea states. Unlike when trained on only one sea state, both NN models perform roughly equally for all sea states.

highest value of 0.1674 occurring for the sea state 6 data and the lowest value of 0.1153 occurring for the sea state 2 data.

By including the additional sea state data in training both NN models were able to perform consistently for all of the sea states that were included in training. However, the NN NARX performance significantly decreased on sea states 2 through 5 and marginally increased on sea state 6. In comparison the GRU Autoencoder saw marginal increases in performance for the low sea states while also gaining large increases on the higher sea states. The GRU Autoencoder demonstrates a significant advantage of the NN NARX model for predicting across varying sea states, which agrees with the results presented in Fig. 5. As the data used for training is normalized applications will require some amount of initialization in order to measure enough data to calculate the variance that should be used to normalize the measured data. Once the variance is calculated the NN models will be ready for use.

C. The Impact of Ship Model

The NN models from Sec. III were trained using data from simulations of a 30m. vessel. Since applications may require multiple or different ship models which could have significantly different dynamics it is important to understand how changing ship models will impact the NN models.

Using ShipMo3D a second dataset was created using a 100m frigate. As the 100m frigate is much larger than the 30m vessel and is not expected to move much in sea state 2 the NN NARX and GRU Autoencoder models are retrained using data from the 30m vessel in sea state 4. Table II shows the testing MSE values for the NN NARX model and the GRU Autoencoder model for the 30m vessel used to create the training data and the new data from the 100m vessel, all at sea state 4. The NN NARX had testing MSE values of 0.5912 and 3.976 for the 30m and 100m vessels respectively while the GRU Autoencoder

		Vessel Size	
Model	Training Data	30m	100m
GRU Autoencoder	30m only	0.1458	0.9483
	30m & 100m	0.1524	0.01156
NN NARX	30m only	0.5912	3.976
	30m & 100m	0.3782	0.1489

TABLE II

THE NN MODELS ARE TRAINED ON DATA FROM SIMULATIONS OF A 30M VESSEL. THE GRU AUTOENCODER MODEL, SHOWN IN RED AS THE LEFT COLUMN, OUT PERFORMS THE NN NARX, SHOWN IN BLUE AS THE RIGHT HAND COLUMN, FOR ALL SEA STATES. AS IS EXPECTED, BOTH NN MODELS PERFORM BETTER ON DATA THAT REFLECTS THE TRAINING DATA THAN ON DATA THAT DOES NOT.

had testing MSE values of 0.1458 and 0.9483 for the 30m and 100m vessels respectively. As demonstrated in Sec. III the GRU Autoencoder model continues to outperform the NN NARX model. The results match expectations and the NN models perform better on the 30m vessel than the 100m vessel.

Another set of NN models were trained using data from both ship model datasets as part of the training and the results are also shown in Table II. The NN NARX had testing MSE values of 0.3782 and 0.1489 for the 30m and 100m vessels respectively. The GRU Autoencoder model outperformed the NN NARX with testing MSE values of 0.1524 and 0.01156 on the 30m and 100m vessels respectively. Unlike in Sec. IV-A and Sec. IV-B the NN models did not generalize in a way that balanced performance between the training datasets. Instead, training optimized the loss by focusing on optimizing predictions from the slower moving 100m vessel. However only the GRU Autoencoders ability to predict the 30m vessel motions was lowered,indicated by the slight rise of 0.0066 in testing MSE. The NN NARX improved on its ability to predict ship motion, indicated by its decrease in testing MSE of 0.2130 and 3.827 for the 30m and 100m vessels, respectively. The GRU Autoencoder saw a decrease in testing MSE of 0.9368 when predicting the 100m vessel.

Overall, the GRU Autoencoder performance remains superior performing model in comparison to the NN NARX model. However, as suggested by the near equal 30m vessel testing MSE values in Table II, there may be no advantage to training using multiple ship models unless the application requires it.

REFERENCES

[1] I. A. Martin, R. A. Irani, A generalized approach to anti-sway control for shipboard cranes, Mechanical Systems and Signal Processing 148 (2021) 107168. doi:https://doi.org/10.1016/j.ymssp.2020.107168.
URL http://www.sciencedirect.com/science/article/pii/S0888327020305549

[2] S. Abujoub, J. Mcphee, C. Westin, R. Irani, Unmanned aerial vehicle landing on maritime vessels using signal prediction of the ship motion, OCEANS 2018 MTS/IEEE Charleston (2018) 1–9doi:10.1109/OCEANS.2018.8604820.

[3] S. Abujoub, J. McPhee, R. A. Irani, Methodologies for landing autonomous aerial vehicles on maritime vessels, Aerospace Science and Technology 106 (2020) 106169. doi:https://doi.org/10.1016/j.ast.2020.106169.
URL http://www.sciencedirect.com/science/article/pii/S1270963820308518

[4] C. K. Tan, J. Wang, Y. C. Paw, F. Liao, Autonomous ship deck landing of a quadrotor using invariant ellipsoid method, IEEE Transactions on Aerospace and Electronic Systems 52 (2) (2016) 891–903.

[5] K. McTaggart, Shipmo3d version 3.0 user manual for computing ship motions in the time and frequency domains, Defense Research and Development Canada - Atlantic (DRDC - Atlantic). Dartmouth, Nova Scotia. TM (2011) 2011–308.

[6] J. McPhee, R. Irani, On-line determination of a go-nogo state using a continous estimation of the system response, in: The Canadian Society for Mechanical Engineering International Congress 2018, 2018. doi:10.25071/10315/35395.

[7] C. R. Bolander, D. F. Hunsaker, A sine-summation algorithm for the prediction of ship deck motion, in: OCEANS 2018 MTS/IEEE Charleston, 2018, pp. 1–10.

[8] H. Jiang, S. Duan, L. Huang, Y. Han, H. Yang, Q. Ma, Scale effects in ar model real-time ship motion prediction, Ocean Engineering 203 (2020) 107202. doi:https://doi.org/10.1016/j.oceaneng.2020.107202.
URL http://www.sciencedirect.com/science/article/pii/S0029801820302596

[9] X. Yang, M. Garratt, H. Pota, Monotonous trend estimation of deck displacement for automatic landing of rotorcraft uavs, Journal of Intelligent Robotic Systems 61 (2011) 267–285. doi:10.1007/s10846-010-9474-z.

[10] F. J. Montáns, F. Chinesta, R. Gómez-Bombarelli, J. N. Kutz, Data-driven modeling and learning in science and engineering, Comptes Rendus Mécanique 347 (11) (2019) 845 – 855, data-Based Engineering Science and Technology. doi:https://doi.org/10.1016/j.crme.2019.11.009.
URL http://www.sciencedirect.com/science/article/pii/S1631072119301809

[11] A. Khan, C. Bil, K. E. Marion, Ship motion prediction for launch and recovery of air vehicles, in: Proceedings of OCEANS 2005 MTS/IEEE, 2005, pp. 2795–2801 Vol. 3.

[12] P. Moriarty, R. Sheehy, P. Doody, Neural networks to aid the autonomous landing of a uav on a ship, in: 2017 28th Irish Signals and Systems Conference (ISSC), 2017, pp. 1–4.

[13] G. Li, B. Kawan, H. Wang, H. Zhang, Neural-network-based modelling and analysis for time series prediction of ship motion, Ship Technology Research 64 (1) (2017) 30–39. doi:10.1080/09377255.2017.1309786.

[14] G. De Masi, F. Gaggiotti, R. Bruschi, M. Venturi, Ship motion prediction by radial basis neural networks, in: 2011 IEEE Workshop On Hybrid Intelligent Models And Applications, 2011, pp. 28–32.

[15] Y. Chu, G. Li, H. Zhang, Incorporation of ship motion prediction into active heave compensation for offshore crane operation, in: 2020 15th IEEE Conference on Industrial Electronics and Applications (ICIEA), 2020, pp. 1444–1449. doi:10.1109/ICIEA48937.2020.9248283.

[16] M. I. Jordan, Attractor dynamics and parallelism in a connectionist sequential machine, Proceedings of the 8th Annular Conference Cognitive Science Society (1986) 531–546.

[17] J. L. Elman, Finding structure in time, Cognitive Science 14 (2) (1990) 179–211. doi:10.1207/s15516709cog1402_1.

[18] Y. Shen, M. Xie, Ship motion extreme short time prediction of ship pitch based on diagonal recurrent neural network, in: Journal of Marine Science and Application, 2005.

[19] J. P. DeCruyenaere, H. M. Hafez, A comparison between kalman filters and recurrent neural networks, in: [Proceedings 1992] IJCNN International Joint Conference on Neural Networks, Vol. 4, 1992, pp. 247–251 vol.4.

[20] B. Y. Suprapto, B. Kusumoputro, A comparison of back propagation neural network and elman recurrent neural network algorithms on altitude control of heavy-lift hexacopter based on direct inverse control, in: 2018 International Conference on Electrical Engineering and Computer Science (ICECOS), 2018, pp. 79–84.

[21] F. Nakhaei, M. Irannajad, Application and comparison of rnn, rbfnn and mnlr approaches on prediction of flotation column performance, International Journal of Mining Science and Technology 25 (6) (2015) 983 – 990. doi:https://doi.org/10.1016/j.ijmst.2015.09.016.
URL http://www.sciencedirect.com/science/article/pii/S2095268615001627

[22] K. M. Rashid, J. Louis, Times-series data augmentation and deep learning for construction equipment activity recognition, Advanced Engineering Informatics 42 (2019) 100944. doi:https://doi.org/10.1016/j.aei.2019.100944.
URL http://www.sciencedirect.com/science/article/pii/S1474034619300886

[23] S. Hochreiter, The vanishing gradient problem during learning recurrent neural nets and problem solutions, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 6 (1998) 107–116. doi:10.1142/S0218488598000094.

[24] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (1997) 1735–80. doi:10.1162/neco.1997.9.8.1735.

[25] G. Zhang, F. Tan, Y. Wu, Ship motion attitude prediction based on an adaptive dynamic particle swarm optimization algorithm and bidirectional lstm neural network, IEEE Access 8 (2020) 90087–90098. doi:10.1109/ACCESS.2020.2993909.

[26] K. Cho, B. van Merriënboer, C. Gulcehre, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using rnn encoder-decoder for statistical machine translation, Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (06 2014). doi:10.3115/v1/D14-1179.

[27] J. Chung, C. Gulcehre, K. Cho, Y. Bengio, Empirical evaluation of gated recurrent neural networks on sequence modeling, NIPS 2014 Workshop on Deep Learning (12 2014).

[28] Y. Su, J. Lin, D. Zhao, C. Guo, C. Wang, H. Guo, Real-time prediction of large-scale ship model vertical acceleration based on recurrent neural network, Journal of Marine Science and Engineering 8 (10 2020). doi:10.3390/jmse8100777.

[29] I. Sutskever, O. Vinyals, Q. V. Le, Sequence to sequence learning with neural networks, in: Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2, NIPS'14, MIT Press, Cambridge, MA, USA, 2014, p. 3104–3112. doi:10.5555/2969033.2969173.

[30] S. H. Park, B. Kim, C. M. Kang, C. C. Chung, J. W. Choi, Sequence-to-sequence prediction of vehicle trajectory via lstm encoder-decoder architecture, in: 2018 IEEE Intelligent Vehicles Symposium (IV), 2018, pp. 1672–1678.

[31] K. McTaggart, Validation of shipmo3d version 3.0 user applications for simulation ship motions, Defense Research and Development Canada - Atlantic (DRDC - Atlantic). Dartmouth, Nova Scotia. TM (2012) 2011–306.

[32] North Atlantic Treaty Organization, (NATO), Standardized wave and wind environments and shipboard reporting of sea conditions, Tech. Rep. STANAG No 4194, NATO STANAG No 4194 (April 1983).

[33] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, TensorFlow: Large-scale machine learning on heterogeneous systems, software available from tensorflow.org (2015).
URL https://www.tensorflow.org/