A Deep Learning Approach to Detect and Isolate Thruster Failures for Dynamically Positioned Vessels Using Motion Data

Peihua Han, Guoyuan Li, *Senior Member, IEEE*, Robert Skulstad, Stian Skjong, and Houxiang Zhang, *Senior Member, IEEE*

***Abstract*—Vessels today are being fully monitored thanks to the advance of sensor technology. The availability of data brings ship intelligence into great attention. As part of ship intelligence, the desire of using advanced data-driven methods to optimize operation also increases. Considering ship motion data reflects the dynamic positioning performance of the vessels and thruster failure might cause drift-offs, it is possible to detect and isolate potential thruster failure using motion data. In this paper, thruster failure detection and isolation are considered as a time series classification problem. A convolutional neural network (CNN) is introduced to learn the mapping from the logged motion sequence to the status of the thruster. CNN is expected to generate task-specific features from the original time series sensors data and then perform the classification. The dataset is collected from a professional simulator in the Offshore Simulation Centre AS. Experiments show that the proposed method can detect and isolate failed thrusters with up to 95% accuracy. The proposed model is further extended to deal with thruster failure in a real-time manner.**

***Index Terms*—Thruster failure, fault detection and isolation, convolutional neural network, dynamic positioning.**

I. Introduction

THE interest of remotely operated and autonomous ships is growing in the maritime industry. With the installation of condition monitoring technology, the operation data of the vessel becomes available, which encourages the development of onboard support systems [1]. How to utilize the data to make the vessel more intelligent and efficient, e.g., assessing the asset's health or optimizing maintenance decisions, is one of the major tasks. Leveraging the statistical or data-driven methods to explore the data could be beneficial and there is increasing attention in this area.

Thrusters are the main propulsion units used to position a modern vessel. To mitigate the effects of thruster failures in dynamic positioning (DP) operation, vessels today with DP classes 2 and 3 [2] have been equipped with redundant thrusters. Once a thruster failure is correctly located, a warning can be sent to a crewmember or a high-level controller and the over-actuated vessel can still maintain its position or perform certain tasks if proper reallocation of the desired thrust is initiated.

Corresponding author: Guoyuan Li, e-mail:guoyuan.li@ntnu.no.

Peihua Han, Guoyuan Li, Robert Skulstad and Houxiang Zhang are with the Department of Ocean Operations and Civil Engineering, Norweigian University of Science and Technology, Aalesund, Norway.

Stian Skjong is with SINTEF Ocean, Trondheim, Norway.

If the thruster failure causes insufficient driving power of the vessels to compensate environmental effects, drift-offs will occur. In such a case, ship motion is correlated to the thruster failure event. Therefore the motion data can be used inversely to pinpoint the failed thruster. The use of motion data to perform thruster fault detection and isolation (FDI) usually involves model-based and data-driven methods. The model-based methods require the development of a mathematical model of ship motion utilizing domain knowledge. Fault diagnosis algorithms can be developed by monitoring the consistency between the measured outputs of the real system and the model outputs [3]. The difference can be represented as residuals and it can be obtained through observer-based methods, parity space, and parameter estimation [4]. The residuals can be used to detect and isolate thruster faults through hypothetical tests. However, the residuals are difficult to design for fault isolation since one residual should be only sensitive to one fault type. In addition, these methods rely on an accurate dynamic model of vessel, which are difficult to develop due to the randomness and complexity of environmental effects.

The data-driven methods treat the problem as pattern recognition or classification problem. This approach has been applied to many component-level fault diagnosis areas such as bearings [5], turbine blades [6] and engines [7]. In general, features can be extracted from the time or frequency domains and then a classical machine learning algorithm such as logistic regression (LR), support vector machine (SVM) and random forest (RF) can be applied. These methods do not require explicit mathematical models but a huge amount of historical data for training. Recently, the possibility of building data-driven predictive models for maritime components such as power plant [8], [9] and diesel engine [10] have been demonstrated. The feature extraction process is a fundamental part and the performance of these methods highly relies on the quality of the extracted features. Such features are demanded for manually designing to characterize the system current state [11]. However, designing such features might be difficult and it requires lots of trial and error. Recently, deep learning has emerged as a potential data-driven method and has achieved state-of-the-art results in image classification [12] and speech recognition [13]. The advantage of deep learning is that it combines hierarchical feature extraction and classification, which suggests that it can automatically learn the representation from raw sensor data and therefore reduce the effort of hand-crafted feature extraction. Therefore, it is promising to perform thruster FDI using a deep learning model.

In this paper, the thruster FDI is treated as a time series classification problem. This paper is not concerned with the failure modes in a thruster (such as failed gears or seals) but rather intends to locate which thruster has failed in a broader range. A convolutional neural network (CNN) is developed for thruster FDI of a dynamically positioned offshore vessel. The control signals and logged ship motions are used as input to the network. The output is the estimated thruster condition of the vessel. Although there have been researches concerning using ship motion data in neural network for control [14], trajectory prediction [15] and sea state estimation [16], it is the first time that the ship motion data is used in neural network for thruster FDI. This approach does not require a dynamic model of vessel and no feature extraction process is needed since CNN can produce hierarchical representations from the raw data. Validation through a case study of an offshore vessel shows that the proposed method can detect a faulty thruster with accuracy up to 95%. Comparisons with feature-based methods show the effectiveness of the proposed method. The major contributions of this work are as follows:

- Transforming the thruster FDI problem into a time series classification problem. The focal loss is presented to address the imbalanced class.
- A deep CNN is proposed for thruster FDI of dynamically positioned ships. The feature extraction process can be omitted since this model can generate features itself.
- An online detection scheme for this model is proposed for the onboard supporting system.

The overall organization of the paper is as follows. Recent and related work of thruster fault diagnosis and time series classification is introduced in Section II. Section III introduces the proposed methodology and Section IV describes the data and training setting. The experimental results are discussed in Section V. Section VI concludes the paper.

II. Related Work

A. Thruster fault diagnosis

The aims of thruster fault diagnostics include fault detection, isolation, and identification. For identifying the failure modes in a thruster, Capocci et al. [17] measured winding temperature, bus voltage, shaft speed, actual shaft speed, current consumption of the thruster, and used these measurements as inputs to train a two-layer feed-forward neural network to perform the classification. The control signals and the motions of the plant are usually used to detect and isolate faulty thruster. Fonod et al. [18] first applied a bank of 5 nonlinear unknown input observers to confine the fault to a possible subset and further isolate the faulty thruster based on the directional cosine approach for autonomous spacecraft. Fabiani et al. [19] used nonlinear principal component analysis for fault detection and artificial neural network (ANN) for fault isolation for AUVs. Sun et al. [20] used an improved particle filter method to detect the fault and estimate the loss in control forces. For offshore vessels, Benetazzo et al. [21] used parity methods and a Luenberger observer to generate the residuals and then a CUSUM algorithm was implemented to detect and isolate thruster failures. Similarly, Cristofaro and Johansen [22] used an unknown input observer to generate the residuals and applied them to perform thruster fault isolation. The model-based method has been developed rapidly for detecting and isolating thruster failures but it requires an accurate dynamic model and most of the residual generated methods are limited to the linear model. In essence, the control signals and the ship motions form multi-variate time-series data, which can characterize a failing thruster. Therefore it is possible to extract useful information from the control signals and ship motion data for thruster fault detection and isolation.

B. Time series classification

Time series classification (TSC) algorithms have been developed over years. The pioneering work focused on distance-based approaches whose key part is to measure the similarity of two time series. Dynamic time warping (DTW) [23] might be the most notable similarity measurement. The classification is usually done by a k-nearest neighbors (KNN) or SVM with similarity-based kernels. Feature-based methods extract features in the time series such as mean and variance, complex features from trend or spectral analysis, and features from well-known shapelets [24]. A time series bag-of-features (TSBF) [25] can be formed and then a traditional classification algorithm such as logistic regression or random forest can be used. The above methods need heavy crafting on data preprocessing and feature engineering. Recently, efforts have been put into exploiting the deep neural networks to provide end-to-end solutions for TSC. Cui et al. [26] proposed a multi-scale CNN and argued that the 1-D convolutional filter can be regarded as a shapelet. Wang et al. [27] proposed a strong baseline for TSC based on a fully CNN and showed that it achieves premium performance in the UCI time series archive. Karim et al. [28] combined a recurrent neural network and CNN for TSC. Song et al. [29] used the attention model for clinical TSC. In summary, it is possible to take advantage of deep learning to extract patterns from the control signal and ship motion data and therefore to perform thruster FDI.

III. Methodology

A. Network architecture

The general layout of the proposed method and the detail network architecture is shown in Fig. 1. The sensor data is used as input to the model while the conditions of the thrusters are given as outputs. The convolutional layer includes three operations:

$$\begin{aligned} \mathbf{s} &= Conv(\mathbf{x}) \\ \mathbf{s} &= BN(\mathbf{s}) \\ \mathbf{s} &= ReLU(\mathbf{s}) \end{aligned} \tag{1}$$

where $\mathbf{x}$ is the input; $Conv$ represents the convolutional operation and it contains the learnable weight; BN denotes the batch normalization [30] layer which helps to accelerate the training process; $ReLu$ [31] is the activation function.

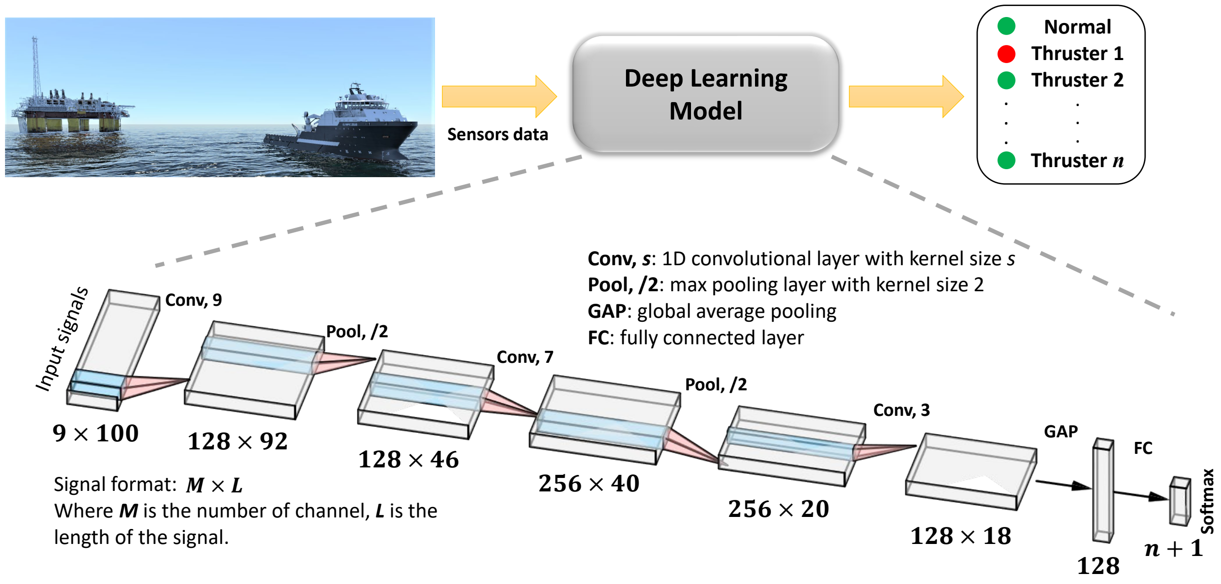

Sensors data
Deep Learning Model
Normal
Thruster 1
Thruster 2
Thruster n
Conv, s: 1D convolutional layer with kernel size s
Pool, /2: max pooling layer with kernel size 2
GAP: global average pooling
FC: fully connected layer
Input signals
Conv, 9
Pool, /2
Conv, 7
Pool, /2
Conv, 3
GAP
FC
Softmax
9×100
128×92
128×46
256×40
256×20
128×18
128
$n+1$
Signal format: $M \times L$
Where M is the number of channel, L is the length of the signal.

Fig. 1: General layout of the method.

Max pooling layers are used after the first two convolutional layers. Global average pooling (GAP) is applied after the final convolutional layer instead of a fully connected layer, which largely reduces the number of weights. Finally, a softmax layer is employed to produce the probability of each class. The convolution layer is fulfilled by three 1-D kernels with the sizes $\{9, 7, 3\}$ and the output channels $\{128, 256, 128\}$, respectively.

Fig. 2 shows a detailed diagram of the proposed method. The model is trained with historical data which also contains scenarios where thrusters are failing. Then it is deployed and provides predictions as new data comes in. Finally, the fault detection and isolation results are confirmed by feeding the network's predictions into a fault detection module. The fault detection functionalities will be illustrated in Section V-E.

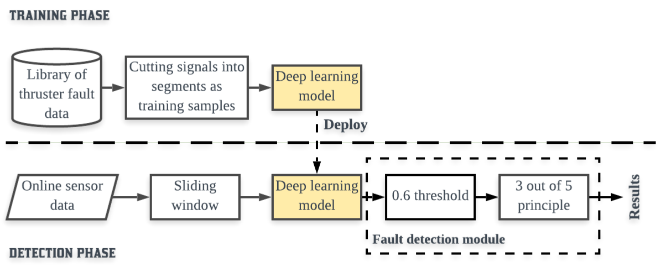


Fig. 2: Block diagram of the method.

B. 1-D convolutional operation

Convolution is a well-established method for handling sequential signals [32]. Suppose that f is a convolutional filter with kernel size s and T is a multi-variate time series with channel number m, the discrete 1-D convolutional operation is defined as:

$$z[i] = \sum_{k=1}^{m} \sum_{j=1}^{s} f_k[j] * T_k[i+j-1] + b \tag{2}$$

where i denotes the i_{th} element of result and b is bias. Fig. 3 illustrates the 1-D convolutional operation process for multivariate time series data. The convolutional filter with size s will move along the time axis with stride length r and repeat the operation as shown in (2). Depending on the filter, the convolution is capable of extracting insightful information from the original time series. For instance, consider a 1-D time series data with a filter $f = [-1, 1]$, the result of the convolution would be the gradient between any two neighboring points. Consider the input time series with size (L_{in}, C_{in}) and filter size (s, C_{out}), the output data size will be (L_{out}, C_{out}).

$$L_{out} = \lfloor \frac{L_{in} - s}{r} + 1 \rfloor \tag{3}$$

where L_{in} and C_{in} is the input length and channel of the data, respectively. C_{out} is the output channel number and it also suggests the number of convolutional filters. When the number and size of the filter is determined, its weight can be learned through the back-propagation algorithm [33].

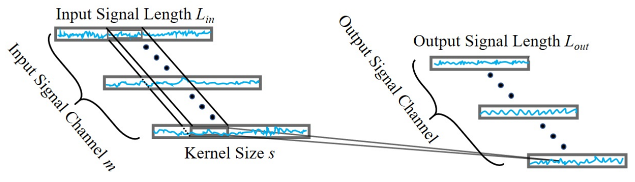


Fig. 3: 1D convolutional operation.

C. Focal loss function

To train the network, the focal loss function [34] is used here instead of the cross-entropy loss function to address the problem of the imbalanced dataset. The focal loss function can be expressed as follows:

$$loss = \sum_{i=1}^{n} -p_i(1-\hat{p}_i)^{\gamma}\log(\hat{p}_i) \tag{4}$$

where $\hat{p}_i$ is the predicted probability value for class i and p_i is the true probability for that class, n denotes the class number, and γ is a tunable parameter. When $\gamma = 0$, this function degrades to the cross-entropy loss function. The term $(1-\hat{p}_i)^{\gamma}$ diminishes the loss assigned to well-classified examples and increases the loss for mis-classified examples. By modulating the cross-entropy loss towards the hard examples, the focal loss is addressed for class imbalance.

IV. Experimental Setting

A. Dataset

The data comes from a commercial simulator developed by the Norwegian company Offshore Simulator Centre AS [35]. The simulator features a simulated environment in which a user may manipulate the wind, waves, and ocean current to mimic environmental conditions. A multi-purpose offshore vessel was selected. This offshore vessel is equipped with 4 tunnel thrusters and 2 main thrusters as presented in Fig. 4. Three different typical sea states are simulated as shown in Table I. The wind, wave, and current will come from the same direction. The direction of the environmental disturbances (α in Fig. 4) is incremented at an interval of 60 degrees from 0 to 360 degrees, relative to the vessel frame. Although smaller intervals will result in less sparse data, which is beneficial to the generalization of the network, we applied an interval of 60 degrees and empirically found that the data satisfies the modeling and analysis.

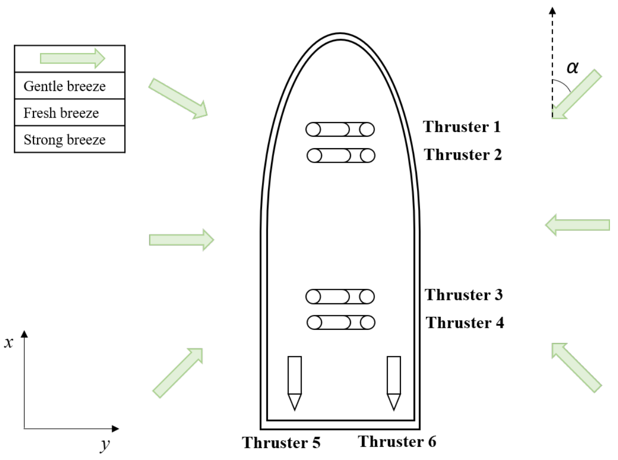


Fig. 4: Thruster configuration.

The DP operation is simulated and the desired position is set to (0, 0). Thrusters are randomly disabled in various environmental conditions. The resulting dataset is shown in Fig. 5. ‘Normal’ denotes no thruster failures and ‘Thruster 1’ represents failure in thruster 1. In total, around 43 hours was simulated whereof 58% without thruster failure. The dataset is relatively unbalanced. Three control signals including the surge, sway, and yaw forces together with the 6 degrees of freedom (DOF) motion data of the vessel were extracted. The data was extracted at a sampling rate of $10\,Hz$.

TABLE I: Descriptions of sea states.

Beaufort scale	Wind velocity (m/s)	Wave height (m)	Current velocity (m/s)
Gentle breeze	4	1	0.2
Fresh breeze	8	2	0.2
Strong breeze	12	3	0.2

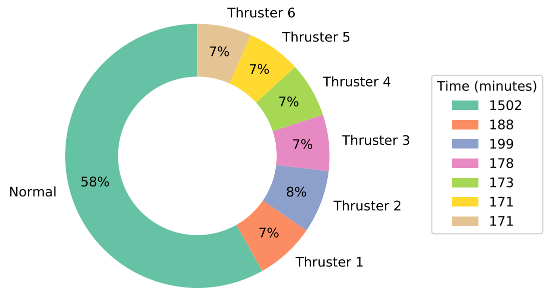


Fig. 5: Thruster failure distribution of the dataset.

B. Measurement noise

In order to provide a more realistic scenario, noise was added to the following measured states:

- Position: The linear position measurements given in the North East Down (NED) frame and angular position including pitch, roll, yaw angle of the vessel.
- Velocity: The three linear velocities and three angular rates.

The position measurements are subjected to a combination of white noise, a bias and a Gauss-Markov (GM) process. The discretized GM process is shown as follows:

$$q[t+1] = \exp(-\frac{\Delta t}{T_c})q[t] + \omega_1[t] \tag{5}$$

where t is the discrete time variable, Δt is the sampling interval, T_c is the correlation time and ω_1 is the Gaussian white noise with a standard deviation of σ. Then the expression for the measured state with the addition of noise term is presented as follows [36]:

$$p[t] = p_{true}[t] + q[t] + \omega_2[t] + \delta \tag{6}$$

where p_{true} is the noiseless measured state, q holds the corresponding GM process noise, ω_2 is the added Gaussian white noise and δ denotes the bias. The angular and linear velocity received only a constant bias and white noise [37]. The parameters for additive noise are shown in Table II.

TABLE II: Parameters used for the additive noise elements of the measured states.

	GM		White noise	Bias
	σ	T_c	σ	δ
Linear position	$0.1\,m$	$240\,s$	$0.2\,m$	[-0.2, 0.2] m
Angular position	$0.1\,^\circ$	$60\,s$	$0.1\,^\circ$	[-0.1, 0.1] $^\circ$
Linear velocity	-	-	$0.05\,m/s$	[-0.01, 0.01] m/s
Angular velocity	-	-	$0.1\,^\circ/s$	[-0.04, 0.04] $^\circ/s$

C. Training settings

The data is split 70%-30% for training and testing. Each input measurement in the training set is normalized with z-score normalization and the corresponding normalization statistics are applied to the test set. The mini-batch size is set to 64 and Adam [38] is selected as the optimizer with a learning rate of 1×10^{-3}. The parameter γ in the focal loss is set to 4. All of the following experiments use the same hyper-parameters and training algorithm settings and there will further elaboration on the effect of these hyperparameters since it is not the focus of this work. The model is trained for 100 epochs. The proposed network is implemented using Pytorch [39].

D. Evaluation metrics

For the multi-class classification problem [40], three evaluation metrics utilizing true positive (TP), true negative (TN) and false positive (FP), are usually used to evaluate the performance of the model. The precision of class i represents the predicted accuracy of this class and is expressed as:

$$P_i = TP_i/(TP_i + FP_i) \tag{7}$$

The recall of class i refers to the percentage of total relevant results correctly classified for this class and is expressed as:

$$R_i = TP_i/(TP_i + TN_i) \tag{8}$$

In order to summarize the models' performance into a single metric, the F1 score is usually used.

$$F1_i = 2 \cdot P_i \cdot R_i/(P_i + R_i) \tag{9}$$

The Macro-Precision, Macro-Recall, and Macro-F1 score are defined as the average precision, recall, and F1 score for all classes.

$$\begin{aligned} P_{macro} &= \frac{1}{n}\sum_{i=1}^{n} P_i \\ R_{macro} &= \frac{1}{n}\sum_{i=1}^{n} R_i \\ F1_{macro} &= \frac{1}{n}\sum_{i=1}^{n} F1_i \end{aligned} \tag{10}$$

Here n is the number of the class. In summary, P_{macro} measures the overall accuracy, R_{macro} measures the overall recall and $F1_{macro}$ is the integration of P_{macro} and R_{macro}.

V. Results and Discussions

A. Input selection

In total 12 ship motion measurements were obtained from the GPS and MRU system for a 6-DOFs vessel, and 3 control signals were obtained from the DP controller. Here we use the following input cases to train the network.

- Case i: All 12 ship motion measurements with the control signals.
- Case ii: 6 ship motion measurements (north and east position, heading angle, surge velocity, sway velocity, yaw velocity) with the control signals.
- Case iii: 6 ship motion measurements (same as case ii) without the control signals.

Fig. 6 shows the variation of $F1_{macro}$ with respect to different input cases and Table III provides the summary after 100 epochs. It can be observed that input case ii performs as well as case i, which indicates that the reduced 6-DOFs motion measurements include enough information for representing the vessel's behaviors in DP operation. Compared with case i and case ii, a significant drop is observed in case iii. This is consistent with the intuition that the motion measurements only contain a certain amount of information, not enough to describe the behaviors of the vessel with the faulty thruster. By combining the 6 ship motion measurements and control signals, the environmental conditions can be reflected by the demanded thrust forces to some extent. Therefore the faulty thruster can be well-isolated by excluding the environmental influence. Moreover, the convergence rate of case iii is lower than those of case i and ii.

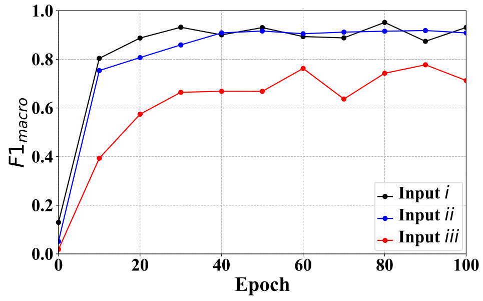


Fig. 6: $F1_{macro}$ for different input cases.

TABLE III: Evaluation on different input cases.

Inputs	P_{macro}	R_{macro}	$F1_{macro}$
case i	0.94	0.92	0.93
case ii	0.91	0.90	0.91
case iii	0.73	0.70	0.71

B. The effect of the window size

Four different windows sizes (100, 200, 300, 400) are investigated. Fig. 7 shows the $F1_{macro}$ and Table IV provides the summary after 100 epochs. It is found that the convergence rates for different window sizes are almost the same and they reach a similar performance. The slight performance decrease in window size 200 might be due to the stochastic gradient descent with mini-batch. A window size of 100 seems to be sufficient for isolating faulty thrusters.

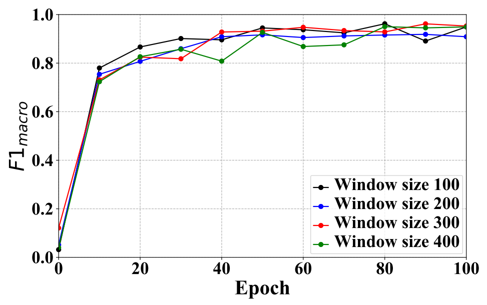


Fig. 7: $F1_{macro}$ for different window sizes.

TABLE IV: Evaluation of different window sizes.

Window size	P_{macro}	R_{macro}	$F1_{macro}$
100	0.95	0.94	0.95
200	0.91	0.90	0.91
300	0.96	0.94	0.95
400	0.96	0.94	0.95

C. Evaluation of different environmental conditions

Here we take input case ii and the window size 100 for further analyses on the performance of the model. Fig. 8 shows the normalized confusion matrix under different sea state levels, where N denotes the label 'normal' and 1 is the label 'thruster 1 failure', etc. It is apparent that it is easy to confuse thruster 1 and 2 but they have an extremely small probability of being classified as other thruster failures. The phenomenon is much more obvious for thruster 3 and 4. The reason might be that they are located close to each other and thus they provide similar functionality for the vessel as shown in Fig. 4. As the sea state level goes from the gentle breeze to the strong breeze, the probability of classifying faulty states to normal states increases. It can be explained by the fact that it is harder for the vessel to keep its position under a stronger breeze. The classification results under these three sea state levels are almost the same; only a small decrease can be found with the increase in sea state level, as shown in Table. V.

TABLE V: Evaluation of different sea states.

Window size	P_{macro}	R_{macro}	$F1_{macro}$
Gentle breeze	0.96	0.95	0.96
Fresh breeze	0.95	0.95	0.95
Strong breeze	0.95	0.94	0.94

In Fig. 4, the six directions of environmental disturbances are further merged into three groups since they are axis-symmetric. Direction 1 is angle of 30° and 330°, direction 2 is angle of 90° and 270°, and direction 3 is 150° and 210°. Fig. 9 shows the confusion matrix for the different directions and Table VI summarizes the results. A similar pattern is shown in Fig. 8 as observed in Fig. 9. For direction 2, thrusters 5 and 6 have a high probability of being mis-classified as normal, but the other thrusters do not. For directions 1 and 3, thrusters 1, 2, 3, and 4 have a higher probability of being mis-classified as normal than for direction 2. The reason might be that tunnel thrusters play more significant parts when environmental disturbances come from direction 2.

TABLE VI: Evaluation on different direction of environmental disturbances.

Window size	P_{macro}	R_{macro}	$F1_{macro}$
Direction 1: $\alpha = 30/330°$	0.96	0.95	0.95
Direction 2: $\alpha = 90/270°$	0.95	0.94	0.94
Direction 3: $\alpha = 150/210°$	0.95	0.95	0.95

D. Comparison with different methods

Two different types of methods are compared here, namely the feature-based method and the end-to-end method (deep learning model). The feature-based method requires data pre-processing while the end-to-end method can work with raw sensor data directly.

For the feature-based method, features are extracted manually from the data. For each sensor, six time domain features and eight frequency features from fast Fourier transform (FFT) and power spectral density (PSD) are extracted. This results in 126 features in total. Down-sampling is performed to reduce the data with label ”normal” to almost the same amount as the faulty data. Three different algorithms are used to train a classifier. The hyper-parameters are selected based on a 3-fold cross-validation grid search. The models are implemented using Scikit-learn [41] in Python.

- Logistic regression (LR): the penalty for l_2 regularization is 2.
- Support vector machine (SVM): RBF kernel function is used. The hyper-parameters is selected as $C = 30$ and $\gamma = 0.01$.
- Random forest (RF): the number of trees is 120, maximum depth of the tree is 20, and the minimum number of samples required to split an internal node is 5.

For the end-to-end method, two models are compared as follows:

- Multilayer perceptron (MLP): a simple feed-forward neural network with 128 hidden units.
- Long short term memory (LSTM): three layers of LSTM contains 128 hidden units are stacked and the final hidden outputs are connected with a Softmax layer.
- Fully convolutional neural network (FCN): the FCN in [27] is adopted for multi-variate time series and the same settings are used.

Table VII presents the performance for different methods. End-to-end methods outperform the feature-based method. Even though it is expected to achieve better results in feature-based methods with a more sophisticated feature extraction process or down-sampling approach, it will be highly time-consuming and laborious. The advantage of end-to-end methods is that they can achieve competitive results without a careful feature extraction process. Moreover, LSTM, FCN and the proposed neural network clearly outperform MLP in this case. The proposed network has a slight advantage over FCN and LSTM. The reason for this might be that the pooling layers in the proposed network reduce the complexity of the network and therefore provide a more generalized result. The comparison also shows that the focal loss can improve the performance of the network.

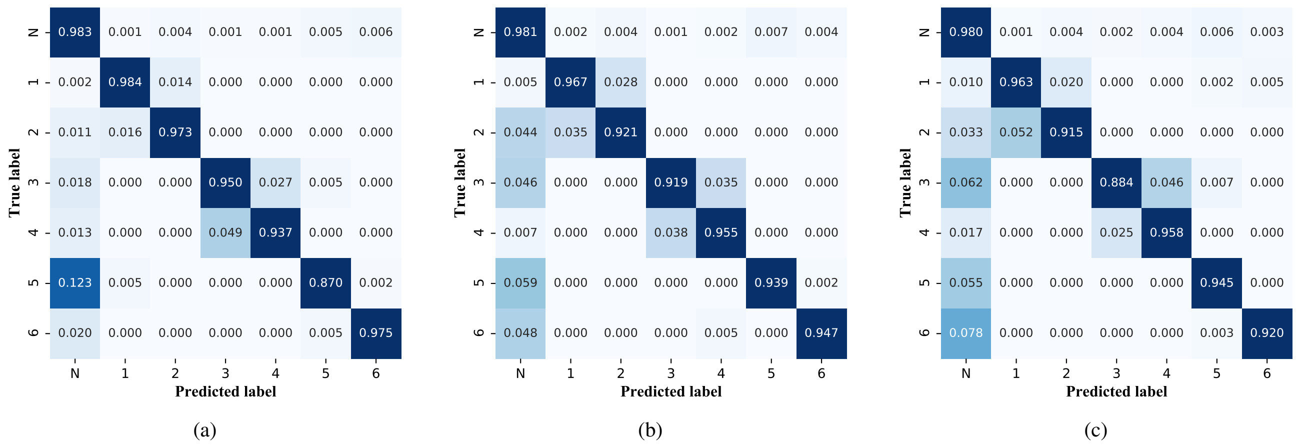


Fig. 8: Confusion matrix under different sea states: (a) Gentle breeze, (b) Fresh breeze, and (c) Strong breeze.

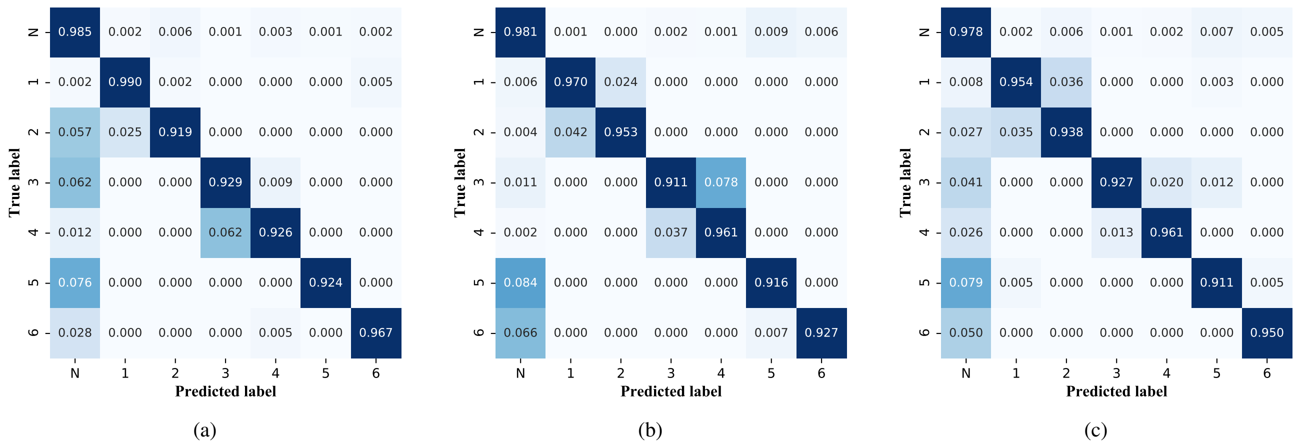


Fig. 9: Confusion matrix under different direction of environmental disturbances: (a) Direction 1, (b) Direction 2, and (c) Direction 3.

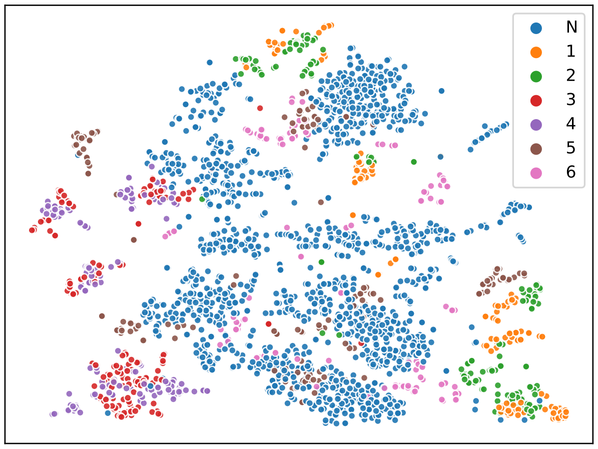


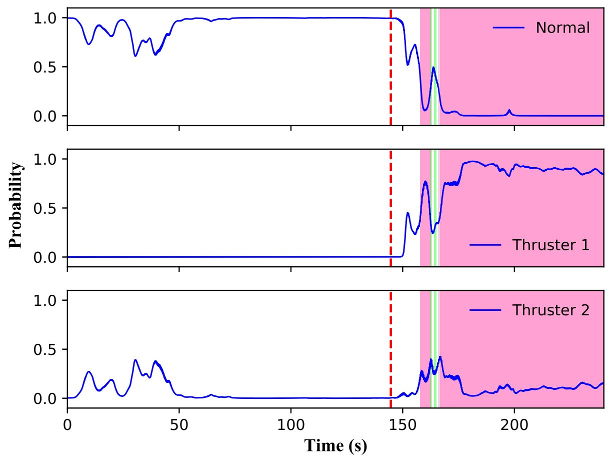
Fig. 10: t-SNE plot of sampled test data in 2D space.

TABLE VII: Comparison of different methods.

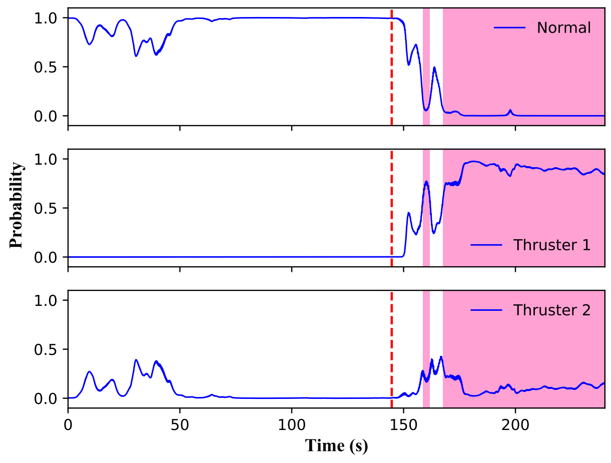
Types	Methods	P_{macro}	R_{macro}	$F1_{macro}$
Feature-based	LR	0.32	0.43	0.35
	SVM	0.56	0.68	0.60
	RF	0.71	0.83	0.75
End-to-end	MLP	0.70	0.81	0.72
	LSTM	0.91	0.91	0.91
	FCN [27]	0.93	0.92	0.93
	Proposed NN (w/o focal loss)	0.93	0.92	0.92
	Proposed NN (focal loss)	0.95	0.94	0.95

In order to illustrate that the proposed network can extract useful features from raw sensor data, the 128 features generated between the GAP layer and the Softmax layer are extracted. Here we get 2000 random samples from the test dataset. The data points in the 128 feature space are then reduced into a 2D space using t-SNE [42] algorithm for better visualization. t-SNE first constructs a probability distribution over pairs of high-dimensional objects and then defines a similar probability distribution over the points in the low-dimensional map by minimizing the Kullback–Leibler divergence. Similar objects are therefore grouped together, which makes it particularly well suited for the visualization of high-dimensional data. Fig. 10 shows the plot of the sampled data, where 'N' denotes normal and '1' is thruster 1 failed, etc. The presence of different groups suggests that the extracted

0018-9456 (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: Norges Teknisk-Naturvitenskapelige Universitet. Downloaded on September 02,2020 at 10:54:45 UTC from IEEE Xplore. Restrictions apply.

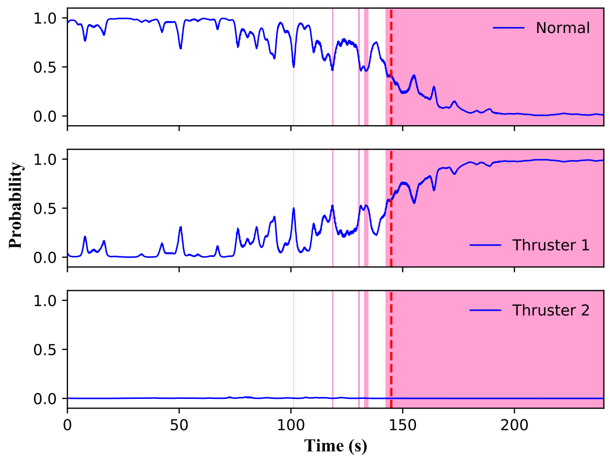


(a)

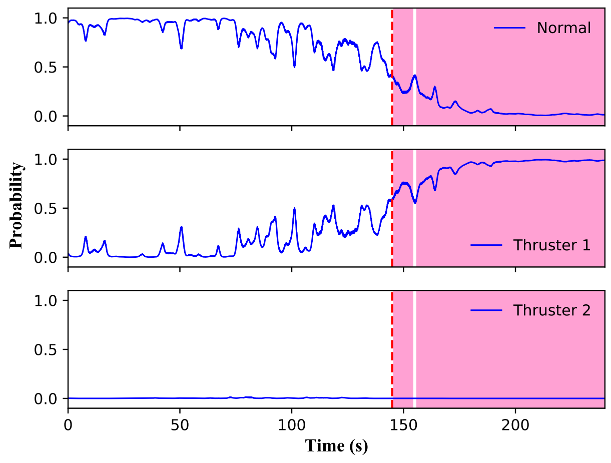


(b)

Fig. 11: Fault isolation under gentle breeze: (a) without fault predictor, and (b) with fault predictor.

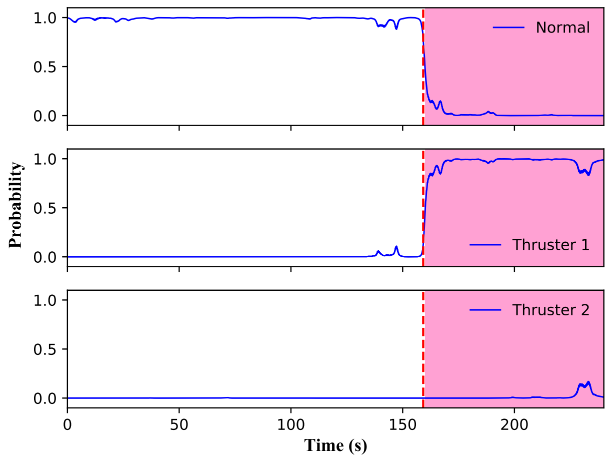


(a)

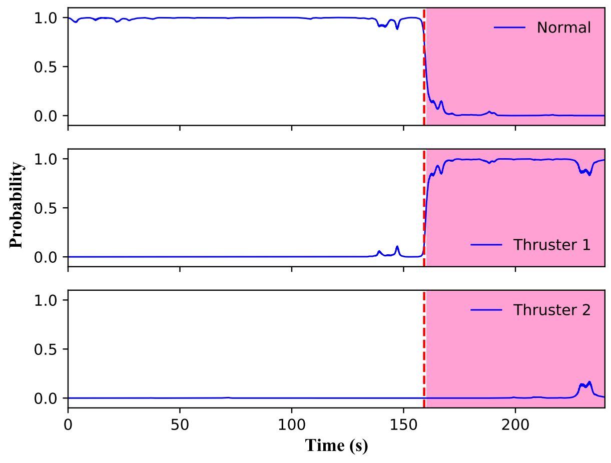


(b)

Fig. 12: Fault isolation under fresh breeze: (a) without fault predictor, and (b) with fault predictor.



(a)



(b)

Fig. 13: Fault isolation under strong breeze: (a) without fault predictor, and (b) with fault predictor.

features from the model are effective to some extent. It can be observed that '1' and '2' are grouped together, '3' and '4' are grouped together, and '5' and '6' are usually grouped with 'N'. This indicates that the proposed network has the ability to extract useful features to isolate faulty thrusters. Since '1' and '2', '3' and '4' are grouped together, which also indicates that these two pairs exhibit a higher probability of mis-classification within the pairs. Moreover, the failures of the two main thrusters '5' and '6' are easy to distinguish from each other but they are both prone to being mis-classified as 'N'.

E. Online detection scheme

To provide onboard support for the vessel, an online detection scheme is proposed. The model is first trained offline as illustrated in the previous section: the historic data is cut into segments with fixed lengths to train a CNN classifier. For the online detection scheme, a sliding window with the same length as the segments is moving as new sensor data are available. Then, the trained CNN classifier provides a predicted probability for each class. Unlike the offline training step, a fault predictor is introduced on the online detection step. A fault is first isolated when its corresponding probability exceeds 0.6. Then a 3/5 principle [43] is applied. As five new data points come in, the sliding window will move forward five times and provide five successive predictions. Only when more than three predictions indicate the same faulty thruster can we confirm the fault identification. Otherwise, it will be considered to be in normal condition. The fault predictor is expected to ensure robustness and eliminate accidental errors.

Fig. 11, Fig. 12 and Fig. 13 present the detection and isolation of thruster fault under gentle, fresh and strong breezes, respectively. The direction of environmental disturbances is set to be 30° with respect to the vessel frame here. The red dashed line indicates the ground truth of the moment when thruster 1 has failed. The background in red represents that the thruster 1 failure is detected while the background in green denotes that the thruster 2 failure is detected. Without the predictor, the result is simply the highest probability of the predicted class. Only the probabilities of normal condition, the thruster 1 failure and the thruster 2 failure are presented since the probabilities of thrusters 3, 4, 5, 6 failure are relatively low in these cases. Table VIII summarizes the detection time and corresponding delays for the three cases.

It can be observed from Fig. 11 that after the thruster 1 has failed, the predicted probability of normal condition decreases while that of thrusters 1 and 2 failure starts to increase. The model provides a nearly 9 seconds delay for detecting the fault and struggles to distinguish between thrusters 1 and 2 in the first place. This is reasonable since the environmental forces are small in this case and therefore the ship motion pattern is relatively unclear. When the environmental forces increase, the pattern is easier to pinpoint, and this can be observed in Fig. 12 under the fresh breeze. The fault predictor enables the model to eliminate the wrong prediction on thruster 2 under gentle breeze and wrong prediction before an actual fault happens under the fresh breeze. In Fig. 13, the impact of the fault predictor is small since the symptoms are obvious. That is why it can be identified with only 1s prediction delay after the failure happens. As shown in Table VIII, the fault can be detected under fresh and strong breezes but has a 9s delay in the gentle breeze. It should be noted that it does not represent the accuracy but a possible prediction delay under different environmental conditions.

TABLE VIII: Summary of the online prediction cases.

	Ground truth time (s)	Failure detection time (s)	Detection delay (s)
Gentle breeze	144.7	153.6	8.9
Fresh breeze	144.9	145.5	0.6
Strong breeze	159.2	160.2	1.0

VI. Conclusion

In this paper, a deep CNN is proposed to detect and isolate potential thruster failures for DP vessels based on the control signals and logged ship motion data. The model is trained with historical data set that contains normal and fault data. The focal loss is used to handle the unbalanced dataset since the amount of normal operation data is much larger than that of fault data. In the simulation cases, the proposed model is able to distinguish thruster failure under various environmental conditions with up to 95% accuracy when being properly trained under the same conditions. Comparisons with the feature-based method show that the model can automatically extract features and performs better. The proposed method provides an approximately 0.2 improvement in F1-score when compared to a simple neural network. An online detection scheme is presented with focus on robust applications. Simulated studies show that the model can predict a failing thruster with negligible delay under the fresh and strong breezes. However, it has a 9 seconds delay under the gentle breeze.

Findings suggest that this method offers good performance for detecting and isolating thruster failures for a DP vessel without requiring any vessel-dependent model. However, as an inherited black-box model, this method suffers from difficulty to interpret its outputs, overfitting, etc. The major drawback of this method is that it is only expected to be reliable when the training data is drawn from the same distribution as the application scenario. In other words, it might not work when encountered with rare environmental conditions which are not included in the training data. Besides real-world fault data is hard to obtain. A reasonable way is to combine simulated data and real-world data. But it also brings new difficulty such as transferring the experience from simulator into the real world. Future research should address these issues and provide a comparison with model-based residual methods.

Acknowledgment

This work was supported by a grant from the Knowledge-Building Project for industry "Digital Twins For Vessel Life Cycle Service" (project no. 28073). The authors would like to thank the technical support from Offshore Simulator Centre AS.

References

[1] DNV GL, "Autonomous and remotely-operated ships," [Online; accessed 16-January-2020]. [Online]. Available: https://www.dnvgl.com/maritime/autonomous-remotely-operated-ships/index.html

[2] D. N. Veritas, "Dynamic positioning system–enhanced reliability dynpos-er," *DNV Guidance*, 2011.

[3] Z. Gao, C. Cecati, and S. X. Ding, "A survey of fault diagnosis and fault-tolerant techniques—part i: Fault diagnosis with model-based and signal-based approaches," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 6, pp. 3757–3767, 2015.

[4] K. Tidriri, N. Chatti, S. Verron, and T. Tiplica, "Bridging data-driven and model-based approaches for process fault diagnosis and health monitoring: A review of researches and future challenges," *Annual Reviews in Control*, vol. 42, pp. 63–81, 2016.

[5] A. Soualhi, K. Medjaher, and N. Zerhouni, "Bearing health monitoring based on hilbert–huang transform, support vector machine, and regression," *IEEE Transactions on Instrumentation and Measurement*, vol. 64, no. 1, pp. 52–62, 2014.

[6] A. Joshuva and V. Sugumaran, "A data driven approach for condition monitoring of wind turbine blade using vibration signals through best-first tree algorithm and functional trees algorithm: A comparative study," *ISA transactions*, vol. 67, pp. 160–172, 2017.

[7] E. Naderi and K. Khorasani, "Data-driven fault detection, isolation and estimation of aircraft gas turbine engine actuator and sensors," *Mechanical Systems and Signal Processing*, vol. 100, pp. 415–438, 2018.

[8] W. Li, A. Monti, and F. Ponci, "Fault detection and classification in medium voltage dc shipboard power systems with wavelets and artificial neural networks," *IEEE Transactions on Instrumentation and Measurement*, vol. 63, no. 11, pp. 2651–2665, 2014.

[9] A. Swider, S. Skjong, and E. Pedersen, "Complementarity of data-driven and simulation modeling based on the power plant model of the offshore vessel," in *ASME 2017 36th International Conference on Ocean, Offshore and Arctic Engineering*. American Society of Mechanical Engineers Digital Collection, 2017.

[10] A. L. Ellefsen, E. Bjørlykhaug, V. Æsøy, and H. Zhang, "An unsupervised reconstruction-based fault detection algorithm for maritime components," *IEEE Access*, vol. 7, pp. 16 101–16 109, 2019.

[11] W. Lu, Y. Li, Y. Cheng, D. Meng, B. Liang, and P. Zhou, "Early fault detection approach with deep architectures," *IEEE Transactions on Instrumentation and Measurement*, vol. 67, no. 7, pp. 1679–1689, 2018.

[12] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[13] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International conference on machine learning*, 2016, pp. 173–182.

[14] G. Li, W. Li, H. P. Hildre, and H. Zhang, "Online learning control of surface vessels for fine trajectory tracking," *Journal of Marine Science and Technology*, vol. 21, no. 2, pp. 251–260, 2016.

[15] R. Skulstad, G. Li, T. I. Fossen, B. Vik, and H. Zhang, "Dead reckoning of dynamically positioned ships: Using an efficient recurrent neural network," *IEEE Robotics & Automation Magazine*, vol. 26, no. 3, pp. 39–51, 2019.

[16] X. Cheng, G. Li, A. L. Ellefsen, S. Chen, H. P. Hildre, and H. Zhang, "A novel densely connected convolutional neural network for sea state estimation using ship motion data," *IEEE Transactions on Instrumentation and Measurement*, 2020.

[17] R. Capocci, E. Omerdic, G. Dooly, and D. Toal, "Fault-tolerant control for rovs using control reallocation and power isolation," *Journal of Marine Science and Engineering*, vol. 6, no. 2, p. 40, 2018.

[18] R. Fonod, D. Henry, C. Charbonnel, and E. Bomschlegl, "A class of nonlinear unknown input observer for fault diagnosis: Application to fault tolerant control of an autonomous spacecraft," in *2014 UKACC International Conference on Control (CONTROL)*. IEEE, 2014, pp. 13–18.

[19] F. Fabiani, S. Grechi, S. Della Tommasina, and A. Caiti, "A nlpca hybrid approach for auv thrusters fault detection and isolation," in *2016 3rd Conference on Control and Fault-Tolerant Systems (SysTol)*. IEEE, 2016, pp. 111–116.

[20] Y.-s. Sun, X.-r. Ran, Y.-m. Li, G.-c. Zhang, and Y.-h. Zhang, "Thruster fault diagnosis method based on gaussian particle filter for autonomous underwater vehicles," *International Journal of Naval Architecture and Ocean Engineering*, vol. 8, no. 3, pp. 243–251, 2016.

[21] F. Benetazzo, G. Ippoliti, S. Longhi, and P. Raspa, "Advanced control for fault-tolerant dynamic positioning of an offshore supply vessel," *Ocean Engineering*, vol. 106, pp. 472–484, 2015.

[22] A. Cristofaro and T. A. Johansen, "Fault tolerant control allocation using unknown input observers," *Automatica*, vol. 50, no. 7, pp. 1891 – 1897, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0005109814001824

[23] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series." in *KDD workshop*, vol. 10, no. 16. Seattle, WA, 1994, pp. 359–370.

[24] L. Ye and E. Keogh, "Time series shapelets: a new primitive for data mining," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2009, pp. 947–956.

[25] M. G. Baydogan, G. Runger, and E. Tuv, "A bag-of-features framework to classify time series," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 11, pp. 2796–2802, 2013.

[26] Z. Cui, W. Chen, and Y. Chen, "Multi-scale convolutional neural networks for time series classification," *arXiv preprint arXiv:1603.06995*, 2016.

[27] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," in *2017 international joint conference on neural networks (IJCNN)*. IEEE, 2017, pp. 1578–1585.

[28] F. Karim, S. Majumdar, H. Darabi, and S. Harford, "Multivariate lstm-fcns for time series classification," *Neural Networks*, vol. 116, pp. 237–245, 2019.

[29] H. Song, D. Rajan, J. J. Thiagarajan, and A. Spanias, "Attend and diagnose: Clinical time series analysis using attention models," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[30] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[31] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[32] S. Mallat, *A wavelet tour of signal processing*. Elsevier, 1999.

[33] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," California Univ San Diego La Jolla Inst for Cognitive Science, Tech. Rep., 1985.

[34] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[35] Offshore Simulator Centre AS, "Offshore Simulator Centre AS." [Online]. Available: http://www.osc.no

[36] T. H. Bryne, "Nonlinear observer design for aided inertial navigation of ships," 2017.

[37] R. Mahony, T. Hamel, P. Morin, and E. Malis, "Nonlinear complementary filters on the special linear group," *International Journal of Control*, vol. 85, no. 10, pp. 1557–1573, 2012.

[38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[39] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[40] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[41] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[42] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[43] B. Yang, R. Liu, and E. Zio, "Remaining useful life prediction based on a double-convolutional neural network architecture," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 12, pp. 9521–9530, 2019.