Gupta *et al. Journal of Big Data* (2024) 11:85
https://doi.org/10.1186/s40537-024-00937-2

Journal of Big Data

RESEARCH Open Access

Correlation-based outlier detection for ships' in-service datasets

Prateek Gupta[1*], Adil Rasheed[2] and Sverre Steen[1]

*Correspondence:
prateek.gupta@ntnu.no

[1] Department of Marine Technology, Norwegian University of Science and Technology (NTNU), Jonsvannsveien, 7050 Trondheim, Sør-trondelag, Norway
[2] Department of Engineering Cybernetics, Norwegian University of Science and Technology (NTNU), O. S. Bragstads plass, 7034 Trondheim, Sør-trondelag, Norway

Abstract

With the advent of big data, it has become increasingly difficult to obtain high-quality data. Solutions are required to remove undesired outlier samples from massively large datasets. Ship operators rely on high-frequency in-service datasets recorded onboard the ships for monitoring the performance of their fleet. The large in-service datasets are known to be highly unbalanced, making it difficult to adopt ordinary outlier detection techniques, as they would also result in the removal of rare but quite valuable data samples. Thus, the current work proposes to establish a correlation-based outlier detection scheme for ships' in-service datasets using two well-known dimensionality reduction methods, namely, Principal Component Analysis (PCA) and Autoencoders. The correlation-based approach detects samples which do not fit the prominent correlations present in the dataset and avoids misidentifying the rare but correlation-following samples in the sparse regions of data domain. The study also attempts to provide the physical meaning of the latent variables obtained using PCA. The effectiveness of the proposed methodology is proven using an actual dataset recorded onboard a ship.

Keywords: Outlier detection, Ship in-service data, PCA, Autoencoders, Non-linear transformations, Latent variables, Correlation

Introduction

Processing and utilizing big data presents a big challenge in today's modern world. The presence of outlier samples in the data often intensifies this challenge. Although outliers can be simply defined as the samples which stand out as outcasts or erroneous when compared to the population, it is far more challenging to detect them. Fortunately, the problem of outlier detection (sometimes referred to as anomaly detection) has been a topic of research and interest since the 19^{th} century [1], leading to the development of a wide range of outlier or anomaly detection methods and algorithms. Some of these methods are developed and only relevant for specific application domains, while the others are more generic and can be adopted for a variety of applications [2]. It is nevertheless important to devise and demonstrate an outlier detection and handling scheme, which may contain one or more of the readily available outlier detection methods, for a dataset concerning a new application domain.

The formulation of an outlier detection scheme for a new application domain would depend on several factors like the application domain itself, type and source of data, objective of further analysis, etc. Such a scheme is, generally, incorporated in the data pre-processing steps to obtain a clean and ready-to-use dataset. Omitting outlier detection and handling mechanisms may lead to imprecise and unhealthy data analysis as well as inaccurate and misleading results, as suggested by McClelland [3]. Moreover, it is well-known that the performance of most of the data-driven methods deteriorates due to the presence of outliers [4]. Thus, outlier detection and handling should be considered an important step before carrying out any prominent analysis on a dataset.

As aforementioned, the formulation of an outlier detection scheme depends on the type of dataset. In case of a correlated dataset, like the one from ships-in-service, an outlier can be viewed as a sample which is defying the expected correlations by a substantial margin, and therefore, in such a case, it may be termed as a *correlation-defying outlier*, as suggested by Gupta et al. [5]. It may be possible to detect these outliers using one of the well-known statistics or Machine-Learning (ML) based outlier detection techniques. However, most of these techniques detect outliers by taking into account the distribution of the data in high dimensional variable (or feature) space, paying not much attention towards the correlation between the variables. Such a technique may cause more harm than good as it would result in detecting extreme values (like extreme weather observations) as well as rare event samples as outliers, which would result in poor predictions in extreme or rare conditions using the models calibrated on the cleaned datasets, as concluded by Suboh and Aziz [6]. Moreover, in case of an unbalanced dataset, the data samples present in the sparse regions of high dimensional variable space would, probably, also suffer the same fate as extreme or rare events, resulting in the loss of valuable information. Therefore, it is recommended here to use a correlation-based outlier detection scheme for unbalanced correlated datasets, like ship in-service datasets.

A correlation-based outlier detection scheme can be simply implemented using a method which can be used to identify the samples that do not follow the prominent correlations observed in the dataset. Although it may be possible to identify such samples using simple regression or curve fitting techniques (as such samples will result in high residuals when fitted to the model), the presence of such outliers will also degrade the model fitting performance, in turn making it difficult to identify the outliers. Moreover, the *curse of dimensionality* and complex non-linear correlations would make it further difficult to diagnose the outliers using such a technique. However, it may be possible to use the well-known correlation-based dimensionality reduction algorithms, namely, Principal Component Analysis (PCA) and autoencoders, as they would help address some of these challenges, and more so because they have been demonstrated to work effectively in the field of outlier detection, as depicted by Sakurada and Yairi [7].

The aim of the current work is to establish a correlation-based outlier detection scheme for an unbalanced dataset obtained from a ship during operations. Such datasets are known to be highly unbalanced and contain several rare but valuable samples, especially at low ship speeds. In order to detect outliers, PCA and autoencoder models are fitted on a dataset representing the hydrodynamic state of a sea-going ship. The dataset includes variables which are either recorded onboard the ship or obtained from a hindcast (or metocean) weather data repository. Since the problem of ship hydrodynamics

is well-researched from the physics point of view, it is expected that the data variables would follow some non-linear correlations. To account for the non-linearities in the case of PCA (as PCA is a linear model), some non-linear transformations are introduced based on the domain knowledge (also suggested by Gupta et al. [8]). The calibrated PCA model is further used to understand the physical meaning of the latent variables (or Principal Components). Finally, a correlation-based outlier detection scheme is proposed for ship in-service datasets using the dimensionality reduction methods.

The contents of this paper are organized as follows. Section "Literature review: outlier detection (OD)" presents an overview of method selection based on a brief literature survey. Section "Methods" contains the theory and formulation associated with the adopted methods, i.e., PCA and autoencoders. The results and conclusion are presented in sects. "Results and discussion" and "Conclusion", respectively, along with sect. "Data exploration & processing" presenting the dataset used to calibrate the models and produce the results.

Literature review: outlier detection (OD)

Since the advent of advanced digital technologies like big data, Internet of Things (IoT), Machine-Learning (ML), Artificial Intelligence (AI), etc., the topic of outlier or anomaly detection has gained interest many-folds. At the same time, the increasing amount of internet traffic, public surveillance and industrial asset monitoring has created a need to devise methods for detecting and filtering out aberrations (or anomalies) in the collected data as well as recognizing malicious activities autonomously. The field of modern medicine has also taken advantage of the above-mentioned digital technologies and invented several useful methods for ML-assisted disease diagnosis based on anomaly detection algorithms. These factors along with several others have encouraged the research and interest in the field of anomaly detection, resulting in the development of several advanced algorithms. In fact, quite recently, a group of researchers assembled an open source library in Python, called *PyOD*[1] and presented by Zhao et al. [9], containing the implementation for more than 40 different outlier detection algorithms.

The availability of a vast number of methods and open-source libraries, with the corresponding software code, is undoubtedly quite helpful when adopting one or more of these methods for a novel application domain. However, it can be quite challenging to ascertain the best-suited method for the given application. It is, therefore, important to develop a basic understanding of the application domain as well as the long list of anomaly detection methods available at one's disposal. The latter is made further complicated by the vast amount of literature available on these anomaly detection methods, where each new method is presented to be superior to any other known method. This is probably the biggest dilemma in today's ML-based research community. The overwhelming amount of literature and methods induces a state of confusion for the group of researchers trying to adopt an ML-based approach for a new application domain. Therefore, the method selection process presents the first challenge here.

[1] URL: https://pyod.readthedocs.io.

Method selection

According to the learning methodology, all the ML-based methods are, in general, broadly divided into the following 3 categories: *supervised*, *semi-supervised*[2] and *unsupervised* methods. The first two types of anomaly or outlier detection methods require target labels or information regarding the outliers (present in the training dataset) so that the model can learn how to differentiate an outlier from an inlier (or normal sample). The third type, i.e., *unsupervised* outlier detection methods, can learn to detect outliers without the need for any target labels. The current problem focuses on detecting outliers in the onboard recorded data for sea-going ships, which is certainly unlabeled as well as tedious to label manually. Thus, it is best to adopt an *unsupervised* ML method here. Moreover, the authors of *PyOD* reported that the *unsupervised* approach for outlier detection may perform better than the *supervised* and *semi-supervised* approaches in most of the cases [10]. The *unsupervised* outlier detection methods detect outliers based on several different techniques. The most widely used techniques are listed as follows:

(i) *Distribution-based:* These methods detect outliers based on the distribution of the data samples. If the data distribution is a well-known one, simple statistical measures like Z-score, interquartile range (IQR), etc. can be used to detect outliers. Otherwise, empirical distributions and histograms can be employed. Generally, in such methods, a threshold is defined by the user for the chosen measure or statistic. All the samples above this threshold are detected as outliers. Histogram-based Outlier Detection (HBOD; [11]) and Kernel Density Estimation (KDE; [12]) are popularly known histogram- and empirical-distribution-based outlier detection methods.

(ii) *Decision-boundary-based:* These outlier detection methods, generally, fall in the category of classification algorithms. These algorithms detect outliers by drawing a decision boundary around the normal (or inlier) data samples. The decision boundary is, generally, obtained such that the margin between the boundary and the data samples is maximized. One-Class Support Vector Machine (OCSVM), originally proposed by Schölkopf et al. [13], is one of the most popular decision-boundary-based outlier detection methods. Owing to its popularity as well as success in the field of anomaly detection, several modifications have been proposed since its inception, most of them implementing a different shape of the decision boundary or surface (originally proposed as a hyperplane by Schölkopf et al. [13]). OCSVM is quite effective for datasets with complex correlations, resulting in very complex graphical distributions, as they first map the data vectors from the *input space* to a *feature space* using non-linear kernel functions. The optimized decision surface is then obtained in the *feature space*.

(iii) *Decision-tree-based:* A decision tree in ML is implemented as a data splitting or partitioning algorithm. Here, the data is recursively split by a randomly chosen feature (or variable) at a randomly chosen splitting value. The recursive data splitting is performed until the objective is achieved, resulting in a hierarchical tree-like structure. When used in a supervised fashion for classification, the objective or the criterion for stopping recursive splitting is to separate samples belonging to

[2] It is also customary to sometimes replace *semi-supervised* learning with *reinforcement* learning or even introduce *reinforcement* learning as a separate category. However, for the current context, it is sufficient to just define the given 3 categories.

individual classes. For *unsupervised* outlier detection, the objective is to separate outliers from inliers, generally, based on the assumption that the samples in sparse regions of the data domain are outliers. Isolation forest [14] is one of the most popular *unsupervised* decision-tree-based outlier detection methods. Here, each individual sample is split or isolated using lines that are orthogonal to data axes and a higher anomaly score is assigned to the samples which need fewer splits. In other words, the samples which can be isolated very easily are considered to be outliers.

(iv) *Distance-based:* As mentioned above, outliers are sometimes viewed as samples which are isolated or far away from other data samples. Thus, it is possible to calculate the distance between the samples, and thereafter, identify samples which are beyond a certain threshold from other samples. These methods are further subcategorized into clustering and nearest-neighbor (or proximity-based) methods. The former divides the data into clusters, and the latter determines the number of samples in the neighborhood of each data sample. The samples falling outside the clusters or having a very sparse neighborhood are detected as outliers. Clustering is one of the most widely used techniques in data analysis and processing, which led to the development of several advanced clustering algorithms. The most popular ones include k-means [15], density-based spatial clustering of applications with noise (DBSCAN; [16]) and Gaussian mixture models (GMM; [17]). The nearest-neighbor methods, on the other hand, are considered state-of-the-art in the field of outlier detection as they are proven quite effective for datasets with very complex distribution, without the need for projecting the data to a kernel-based feature space (as in the case of OCSVM). Local outlier factor (LOF; [18]) is the most frequently adopted and successful nearest-neighbor outlier detection method.

The weakness of all the above methods lies in the fact that they can, at best, detect outliers based on the location and distribution of data in the multidimensional variable space. None of them understands the correlation between data variables, and therefore, would always result in detecting data samples in sparse regions of the variable space as outliers, even when such samples can be rare and valuable. A correlation-based outlier scheme, discussed below, would understand the prominent correlations in the dataset and detect outliers due to their deviation from these correlation trends.

Correlation-based outlier detection

A dataset generally contains correlated variables, and data samples are expected to depict this correlation. It is, therefore, possible to define outliers based on the fact that outliers would substantially deviate from the correlation observed in the rest of the data. In an *unsupervised* setting, a correlation-based model is calibrated using the complete dataset, and the samples which result in high calibration residuals are detected as outliers. There are two main types of *unsupervised* correlation-based outlier detection methods: Regression and Factorization. The regression methods regresses the data on itself, i.e., the model is calibrated such that the input and target variables are the same. In other words, the model attempts to learn the correlation between the input variables, and thereafter, tries to reconstruct the dataset on the target side. Here, the reconstruction error is minimized to obtain the optimum model, and the final reconstruction error or

residuals are used to assign outlier scores. The samples with high reconstruction residuals are detected as outliers. Neural-network-based autoencoders [19] is the most popularly used *unsupervised* regression method for outlier detection.

Factorization methods, on the other hand, factorize the data matrix into several factors or components, each factor carrying a certain amount of variance (extracted from the data matrix). Here, the idea is to project the input features onto a latent space, characterized by the obtained factors. The factors or components are, in fact, the direction cosines of the latent space. Principal Component Analysis (PCA; [20]) is a frequently used method for factorization-based outlier detection. PCA obtains the components such that the first component carries the maximum variance and the succeeding components carry as much of the remaining variance as possible. Since PCA is a linear method, the outliers can be detected with the help of Hotelling's T^2 statistics [21] and the so-called Q-residuals [22], further explained in sect. "Principal component analysis (PCA)". Hotelling's T^2 statistics represent the leverage of an individual data sample on the model, and the Q-residuals represent the reconstruction residuals. The samples with high leverage and residuals are considered outliers as they deviate significantly from the underlying correlations in the dataset.

The advantage of using a correlation-based method is that it would probably result in better generalization for a correlated dataset. In other words, it may be able to effectively detect outliers even outside the available (or training) data limits. A better generalization would also be advantageous for unbalanced datasets, where the data is not distributed evenly over the complete data domain. Unlike the methods mentioned in the previous section, the correlation-based outlier detection methods would not identify samples located in sparse regions, i.e., rare but valuable samples, as outliers, unless they are defying the expected correlations. The dataset used here is obtained from a sea-going ship, and ship-in-service datasets are generally unbalanced as the ship speed is kept constant for the most part of a voyage. Moreover, a correlation-based method can also be used to study the correlation between the data variables and confirm our physical understanding of the phenomenon. Therefore, correlation-based outlier detection methods, namely, PCA and autoencoders are used here to detect outliers in the current study. One drawback of using correlation-based methods is that the model may develop a bias due to the presence of a large number of outliers (compared to the total number of samples). In such a case, robust versions of PCA and autoencoders can be employed, as suggested by Chalapathy et al. [23].

Methods

Following the arguments presented above, the current work uses correlation-based outlier detection methods, namely, PCA and autoencoders, to detect outliers. Since PCA is a linear method, it is attempted to enhance it by using some non-linear transformations, obtained using the domain knowledge applicable to the dataset used here.

Principal component analysis (PCA)

PCA is an unsupervised ML method, which factorizes the data matrix into orthogonally independent and uncorrelated factors, called Principal Components (PCs). The PCs absorb the variability available in the dataset such that the first PC absorbs the

maximum variance and the subsequent PCs absorb as much of the remaining variance as possible. Consequently, due to the accumulation of the majority of variance in the first few PCs, PCA is helpful in reducing the dimensionality of the dataset. In such a case, the last remaining PCs, containing a small amount of remaining variance, are discarded as noise in the dataset. Thus, PCA splits the data matrix ($\mathbf{X}$) into a factorized or modelled part ($\mathbf{X}_{M,\,A}$) and the noise or residuals ($\mathbf{E}_A$) as follows:

$$\mathbf{X}^{m\times n} = \mathbf{X}^{m\times n}_{M,\,A} + \mathbf{E}^{m\times n}_{A} \tag{1}$$

The superscripts show the dimensions of each term, i.e., $m \times n$ shows that the data matrix ($\mathbf{X}$) has m rows (or samples) and n columns (or variables). The subscript, A, is formally called the dimensionality of the factorized part, i.e., the number of PCs containing the majority of the variance. It should be noted that the data matrix ($\mathbf{X}$) is generally standardized by subtracting the mean and dividing by the standard deviation before factorization, as shown by Gupta et al. [24]. The modelled or factorized part ($\mathbf{X}_{M,\,A}$) can be further written as a dot product of PC scores matrix ($\mathbf{T}_A$) and the transpose of PC loadings matrix ($\mathbf{P}_A^\top$). Here, $\mathbf{P}_A^\top$ represents the transpose of $\mathbf{P}_A$.

$$\mathbf{X}^{m\times n} = \mathbf{T}^{m\times A}_{A}.\,\mathbf{P}_A^{\top\,A\times n} + \mathbf{E}^{m\times n}_{A} = \sum_{i=1}^{A} \boldsymbol{t}_i^{m\times 1}.\,\boldsymbol{p}_i^{\top\,1\times n} + \mathbf{E}^{m\times n}_{A} \tag{2}$$

Here, $\boldsymbol{t}_i$ is a column vector, $\boldsymbol{p}_i^\top$ is a row vector, and i is the PC number. The loadings ($\boldsymbol{p}_i$) represent the orthonormal[3] eigenvectors or direction cosines in the latent or PC space, and the scores ($\boldsymbol{t}_i$) represent the corresponding eigenvalue-associated orthogonal[4] vectors or the location of the data samples in the latent space. The eigenvectors are also called PCs, which are basically just the latent variables. The two most popular algorithms used to estimate the PC scores and loadings are Singular Value Decomposition (SVD; [25]) and Nonlinear Iterative Partial Least Squares (NIPALS; [26]).

Influence Plots

In order to detect outliers using PCA, statisticians generally employ influence plots. The influence plot helps identify outliers based on the residuals ($\mathbf{E}_i$) and the leverage or influence of each data sample on the model. Here, $\mathbf{E}_i$ is the residual matrix left after extracting i PCs from $\mathbf{X}$. The residuals ($\mathbf{E}_i$) are used to calculate the Q-residuals for each sample by simply squaring the standardized residuals,[5] and then, summing over all the variables. In vector notations, the Q-residual corresponding to k^{th} data sample and i^{th} PC is calculated as:

$$Q_{i,\,k} = \boldsymbol{e}_{i,\,k}.\,\boldsymbol{e}_{i,\,k}^\top \tag{3}$$

Here, $\boldsymbol{e}_{i,\,k}$ is the k^{th} row, corresponding to the k^{th} data sample, in the standardized residual matrix ($\mathbf{E}_i.\,\mathbf{S}$, where $\mathbf{S}$ is a diagonal matrix containing the inverse of standard deviations for each data variable), and the symbol $^\top$ represents the transpose. If the data

[3] Orthonormality means that $\boldsymbol{p}_i^\top.\,\boldsymbol{p}_i = 1$.

[4] Orthogonality means that $\boldsymbol{t}_i^\top.\,\boldsymbol{t}_i = \lambda_i$, where λ_i is the eigenvalue associated with the i^{th} PC.

[5] Standardized residuals are obtained after dividing the residuals by the estimate of the standard deviation for the corresponding data variable.

matrix ($\mathbf{X}$, in eq. 2) is standardized, the estimates of standard deviations would be 1 and $\mathbf{S}$ would become an identity matrix. Since the residuals of a linear regression model follow a Gaussian (or normal) distribution, it is possible to obtain a critical limit for the Q-residuals (as presented by Jackson and Mudholkar [22, 27], and recently by Thennadil et al. [28]), which can be further used to detect outliers (demonstrated further in sect. "Outlier detection").

The other axis of an influence plot, i.e., leverage, is quantified as Hotelling's T^2 statistics [21]. The T^2 statistic represents the distance of the data sample from the center (or multivariate mean) of the dataset, and therefore, its influence on the model. Hotelling's T^2 statistics corresponding to k^{th} data sample and i^{th} PC can be calculated as follows:

$$T^2_{i,k} = \boldsymbol{t}_{i,k}.\,\lambda_i^{-1}.\,\boldsymbol{t}_{i,k}^{\top} \tag{4}$$

Where λ_i is the eigenvalue associated with i^{th} PC. Similar to Q-residuals, it is possible to obtain a critical limit for Hotelling's T^2 statistics, as presented by MacGregor and Kourti [29] and recently by Thennadil et al. [28].

Autoencoders (AE)

Autoencoders, originally introduced as Replicator Neural Network (RNN; [19]), is a neural-network-based ML method. A neural network can be seen as a complex mathematical transformation applied to the input data in order to map it to the corresponding target values. Here, the complexity arises due to the fact that the mathematical transformation is actually a series of linear or non-linear mappings (or mathematical operations) applied in succession, depicted by the sequence of layers in the network architecture. The target ($\mathbf{Y}$) can, therefore, be written as a function of the input ($\mathbf{X}$) as follows:

$$\mathbf{Y} = f(\mathbf{X};\Theta) = f^L_{\theta_L}(f^{L-1}_{\theta_{L-1}}(..f^2_{\theta_2}(f^1_{\theta_1}(\mathbf{X})))) \tag{5}$$

Here, $\Theta = [\theta_1, \theta_2, ..., \theta_{L-1}, \theta_L]$ is the set of hyperparameters, and L is the number of layers in the network architecture. Further, each mapping ($f^l_{\theta_l}$) is defined as a linear or non-linear transformation applied after a linear operation.

$$f^l_{\theta_l}(\mathbf{X}) = \sigma_l(\mathbf{W}_l \cdot f^{l-1}_{\theta_{l-1}}(\mathbf{X}) + \mathbf{b}_l) \tag{6}$$

Here, σ_l, $\mathbf{W}_l$ and $\boldsymbol{b}_l$ are the hyperparameters of the l^{th} layer in the model (constituting θ_l), called activation function, weight matrix and bias vector, respectively. The non-linearity in the model is introduced by using a non-linear activation function, which can be the same or different in different layers of the model. The most frequently used activation functions are linear, sigmoid, ReLU (Rectified Linear Unit) and tanh (hyperbolic tangent). The activation functions are user-specified, but the other hyperparameters, i.e., weights and biases, can be estimated by minimizing the cost function during model training, considering the hypothesis $\mathbf{Y} = f(\mathbf{X};\Theta)$. The most commonly used cost or loss function is Mean Squared Error (MSE) with L_2 regularization, as presented below:

$$\mathbf{J}(\Theta) = \frac{1}{m}(\hat{\mathbf{Y}} - \mathbf{Y})^{\top}(\hat{\mathbf{Y}} - \mathbf{Y}) + \lambda \sum_{l=1}^{L}(||\mathbf{W}_l||^2 + ||\boldsymbol{b}_l||^2) \tag{7}$$

Where m is the number of samples in $\mathbf{Y}$, $\hat{\mathbf{Y}}$ is the estimate of $\mathbf{Y}$ obtained by the model and λ is the weight decay or regularization parameter.

An autoencoder is an adaptation from an ordinary Artificial Neural Network (ANN) with the following main difference:

(i) ANN is a supervised ML method, requiring target or labeled data, whereas autoencoders is an unsupervised ML method, where the input itself is used as the target, resulting in its original name, Replicator Neural Network [19].

(ii) Unlike ANN, autoencoders must have an odd number of layers in the model architecture, where the neurons in the center-most layer are analogous to factors or PCs in the case of factorization methods like PCA. The simplest autoencoder model contains 3 layers, i.e., input, target and a hidden layer. Moreover, the center-most hidden layer generally contains a smaller number of neurons as compared to the number of input features.

Results and discussion

The primary objective of the current paper is to demonstrate correlation-based outlier detection using dimensionality reduction methods, namely, PCA and autoencoders. However, it may be a good idea to first understand how these methods work. It can be clearly understood from sect. "Literature review: outlier detection (OD)" that PCA obtains a set of optimized latent variables (popularly called PCs) to reduce the dimensionality of the data, and even though it is not that easy to realize, autoencoders do the same. In the latter case, the neurons in the center-most layer (generally containing the smallest number of units) represent the latent variables. The primary difference between PCA and autoencoders is, therefore, just the fact that the latent variables from PCA are orthogonal to each other, whereas they are non-orthogonal in the case of autoencoders, which makes a lot of difference as shown further. In order to demonstrate this, sect. "Latent variables (LVs)" presents the latent variables obtained using both these algorithms, based on the dataset described in the following section ("Data exploration & processing"). The latent variables obtained using PCA are further studied to understand their physical meaning (in sect. "Physical meaning of LVs"). Further, sect. "Data reconstruction" presents the data reconstructed using a selected set of latent variables from both the algorithms, and finally, sect. "Outlier detection" presents the results for outlier detection using PCA and autoencoders.

Data exploration & processing

The data used for the current work is an assimilation of data recorded onboard 7000 TEU[6] post-panamax container ship and weather hindcast data obtained from one of the metocean data repositories. The onboard recorded data samples are obtained as 15-minute averaged values using an onboard installed energy management web application, called Marorka Online.[7] The data is recorded over a period of about 4.5 years covering numerous sea voyages. The wind and wave hindcast data is interpolated to the ship's location in time for all the onboard recorded data samples. The wave data is

[6] Twenty-foot Equivalent Unit.
[7] www.marorka.com

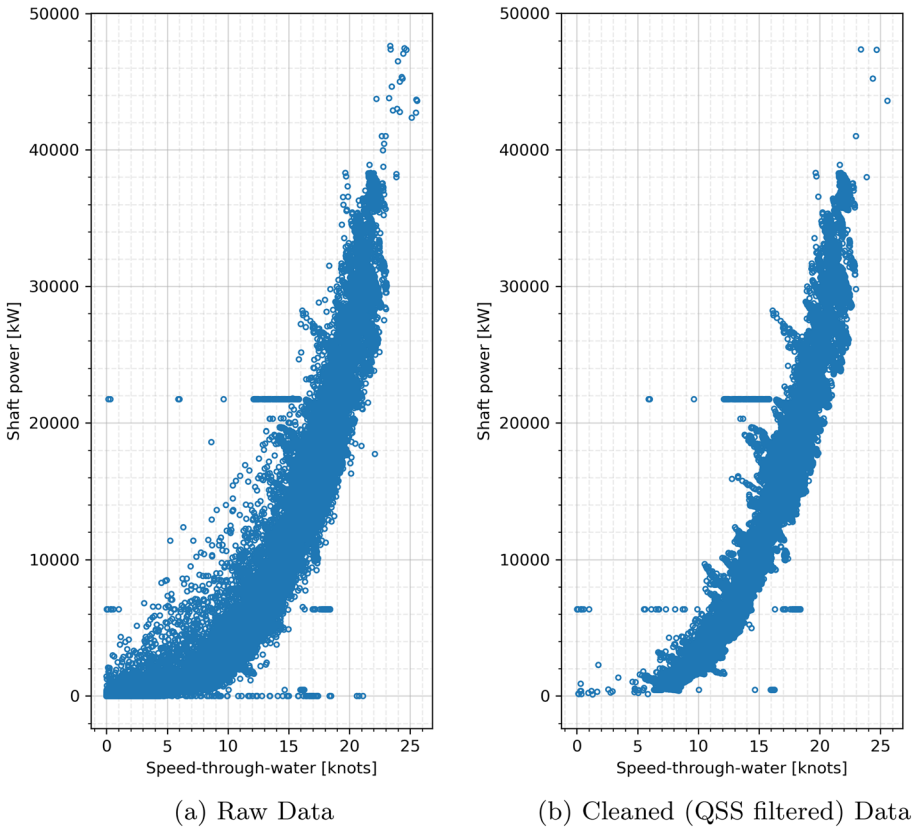


Fig. 1 Speed-through-water vs shaft power data recorded onboard the ship. The raw data contains samples from all the voyages recorded over a period of about 4.5 years. The cleaned data is obtained after applying the quasi-steady-state filter [5] on the raw data

obtained from MFWAM (Météo-France WAve Model),[8] but the source of the wind data is unknown. Unfortunately, no additional information regarding the ship is available as the data is supplied anonymously.

Figure 1 shows the speed-through-water (STW) and shaft power raw data recorded onboard the ship over a period of about 4.5 years. The raw data is further processed as per the data processing framework presented by Gupta et al. [5]. Since the onboard recorded ship data does not include the information regarding the acceleration and deceleration of the ship, the two-stage quasi-steady-state filter suggested by Gupta et al. [5] is used here to remove all the samples where the propeller shaft rpm of the ship is voluntarily changed by the ship's captain. Such samples would be dominated by unaccounted dynamic effects, like non-zero gradients of the ship's motion. Including them for model calibration may result in an undesirably biased model. The right-hand side subplot in Fig. 1 shows the samples remaining after applying the quasi-steady-state filter. These remaining samples are said to be in a quasi-steady state, where the gradients of the ship's motions are negligible.

Looking at the cleaned data (Fig. 1b), it is observed that some samples, which are falling on horizontal straight lines just above the 6000 kW mark and just below the 22,000

[8] www.meteofrance.com

Table 1 Data variables used by ML models. Here, 'All' means all three models, i.e., (linear) PCA, NL-PCA, and Autoencoders, and V_{WL} is used as a symbol for longitudinal wind speed

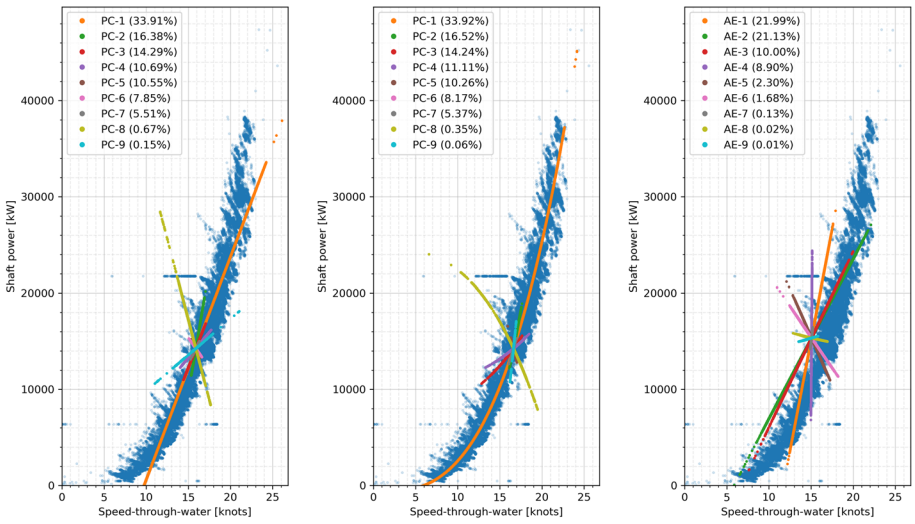
Models	Variables
All	Shaft power, trim-by-aft
	Relative mean wave direction, mean wave period
PCA & Autoencoders	Shaft rpm, speed-through-water, mean draft,
	Longitudinal wind speed (V_{WL}), Significant wave height
NL-PCA	Shaft rpm^3, Speed-through-water^3, Mean draft$^{1/2}$,
	Quadratic longitudinal wind speed ($V_{WL}.\|V_{WL}\|$), Significant wave height^2

kW mark, are depicting strange behavior. Additionally, some more samples at very small shaft power (< 1000 kW) have unexpectedly large STW (around 15 knots). These samples clearly do not follow the correlation depicted by other data samples, and therefore, they can be labeled as *correlation-defying* outliers (as suggested by Gupta et al. [5]). A further investigation reveals that these samples show anomalous behavior due to the temporary freezing of shaft power and rpm sensors during data recording. Nevertheless, such samples should be identified and dealt with before the dataset can be used for any further analysis. Thus, the methodology developed here would be validated based on the criteria that at least these anomalous samples are identified as outliers.

Latent variables (LVs)

As mentioned in sect. "Principal component analysis (PCA)", an LV is nothing but a direction cosine, representing a particular direction in the high dimensional variable space. So, the latent space is in fact same as the original variable space but with a different set of axes, and in the case of linear models like PCA, the new set of axes are actually just rotated and/or translated versions of the original data axes. However, if a non-linear model like autoencoders is used then the axes can be transformed in a much more complicated manner. Similar results can be obtained by applying a set of non-linear transformations to the dataset before carrying out PCA, forming the basis for methods like kernel PCA [30]. Unfortunately, such methods are known to be computationally expensive (as pointed out by Sakurada and Yairi [7]) as well as difficult to understand. However, since the problem of ship hydrodynamics is well-studied, it is possible to use simple non-linear transformations obtained from our domain knowledge to handle non-linearities in the data, as demonstrated by Gupta et al. [8]. Table 1 shows the non-linear transformations used for NL-PCA, i.e., the PCA model with non-linear transformations, as well as the variables (or features) used in the (linear) PCA and autoencoders model.

In the case of PCA, the number of PCs (or LVs) is limited by the rank of the data matrix, i.e., the number of linearly independent columns or variables. Additionally, the latent space is organized such that most of the variance is contained within a very small number of PCs, resulting in dimensionality reduction. However, if most of the variables are linearly independent, which is most definitely true for the current case (observing the list of variables in Table 1), reducing the dimensionality of the data is not lucrative anymore, but studying the obtained PCs to understand the correlations between the data variables is still useful. Figure 2 shows the LVs obtained using the (linear) PCA,



(a) PCA (b) NL-PCA (c) AE

Fig. 2 Latent variables (LVs) obtained from all three models plotted over the cleaned (QSS filtered) data

Table 2 Hyperparameters for the autoencoders model

Sl. No.	Hyper-parameter	Value
1	Optimization algorithm	Stochastic gradient descent (Adam)
2	Loss function	Mean squared error (MSE)
3	Activation function	Hyperbolic tangent (tanh)
4	Batch size	128
5	Weight decay (for Adam optimizer)	10^{-7}
6	No. of hidden layers	3

NL-PCA, and autoencoder models. In the case of PCA, the projection of the i^{th} LV on the variable space (shown in Fig. 2) is obtained by, first, calculating the i^{th} PC matrix ($\mathbf{PC}_i$), as per the following equation, and then, extracting the columns corresponding to the variables (speed-though-water and shaft power in this case) on which the projection is being presented.

$$\mathbf{PC}_i = \boldsymbol{t}_i^{m\times 1}.\boldsymbol{p}_i^{\top\ 1\times n} \tag{8}$$

The LVs from autoencoders, on the other hand, are calculated by dropping (or multiplying by zero) the output from all except for 1 neuron in the center-most hidden layer. The percentage values presented in the legends of Fig. 2 show the explained variance, calculated as the variance of LV divided by the total variance in the data. It should be noted here that the total explained variance of all the LVs from PCA will add to an almost perfect 100%, whereas it will not be the same in case of autoencoders, due to non-orthogonality, resulting in duplication or leaking of variance. The hyperparameters of autoencoders, listed in Table 2, are tuned to obtain the optimum results for the given dataset. However, in order to draw a fair comparison with the PCA models, the number

Table 3 Correlation loadings for NL-PCA model

Sl. No.	Variables	PC-1	PC-2	PC-3	PC-4	PC-5	PC-6	PC-7	PC-8	PC-9
1	Shaft power	0.983	-0.122	0.056	-0.036	0.039	0.020	-0.037	-0.099	0.042
2	Rel. mean wave direction	0.123	0.084	0.358	0.645	-0.653	0.057	0.061	0.000	0.001
3	Mean wave period	0.093	-0.104	-0.704	-0.104	-0.438	-0.530	-0.026	-0.004	0.001
4	Trim-by-aft	-0.176	-0.791	0.268	-0.187	-0.064	-0.100	0.471	-0.002	0.001
5	Shaft rpm^3	0.990	-0.082	0.071	-0.061	0.013	-0.006	-0.006	-0.043	-0.056
6	Speed-through-$water^3$	0.970	-0.075	0.122	-0.130	-0.030	-0.029	-0.033	0.140	0.013
7	Significant wave $height^2$	0.147	-0.285	-0.702	0.118	-0.100	0.608	0.105	0.011	-0.001
8	Mean $draft^{1/2}$	0.272	0.796	-0.168	0.032	0.125	-0.060	0.493	0.000	0.003
9	$V_{WL}.\lvert V_{WL}\rvert$	0.130	-0.316	-0.203	0.708	0.523	-0.259	0.016	0.014	0.000

The scaled values are graphically presented in figure 3

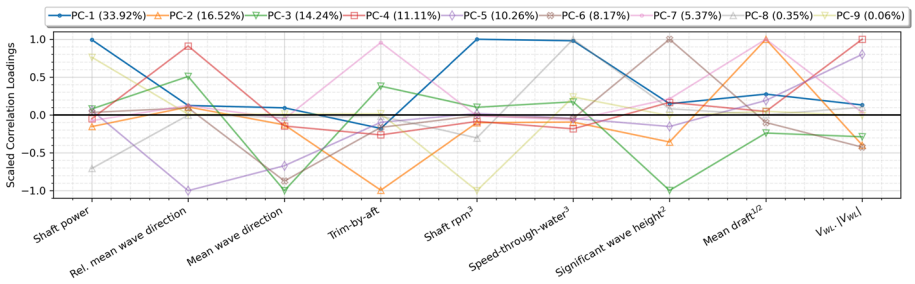


Fig. 3 Scaled correlation loadings for NL-PCA model, based on the values presented in Table 3. The percentage values in the plot legend are the amount of variance or information contained in the corresponding PC

of neurons in all 3 hidden layers is used as 9 only for the results presented in the current section. For further sects. ("Data reconstruction" and "Outlier detection"), the number of neurons in the hidden layers is also tuned. Looking at Fig. 2, as expected, the PCs from the linear PCA model are unable to fit the non-linear trends, whereas the non-linear PCs (from NL-PCA model) clearly fit well to the cubic trends in the data. On the other hand, it is surprising to see that the LVs obtained from the autoencoders model seem mostly linear, even though non-linear activation functions are used in each hidden layer. The results, therefore, indicate that the PCs produced by the NL-PCA model seem most relevant to studying and understanding the prominent correlations within the dataset.

Physical meaning of LVs

The correlation between the data variables and PCs is quantified as correlation loadings, also used by Cadima and Jolliffe [31] to interpret the PCs. The correlation loadings can further be used to: (a) Study the correlation between different data variables; (b) Understand the relative importance of each data variable; and (c) Interpret the physical meaning of each LV. Table 3 presents the correlation loadings for all the PCs obtained using the current dataset. Two (or more) data variables which are strongly correlated with an individual PC (highlighted by the red background in Table 3) intrinsically have a strong correlation with each other. The correlation loadings corresponding to each PC are also shown graphically in Fig. 3 after scaling, such that the highest correlation loading is scaled up to 1 for each PC. Table 3 and Fig. 3 clearly show that the shaft power is highly correlated with the shaft rpm and speed-through-water, whereas the correlation with other variables is almost negligible. This is expected as the data, presented in Fig. 1, shows a very small influence of environmental loads.

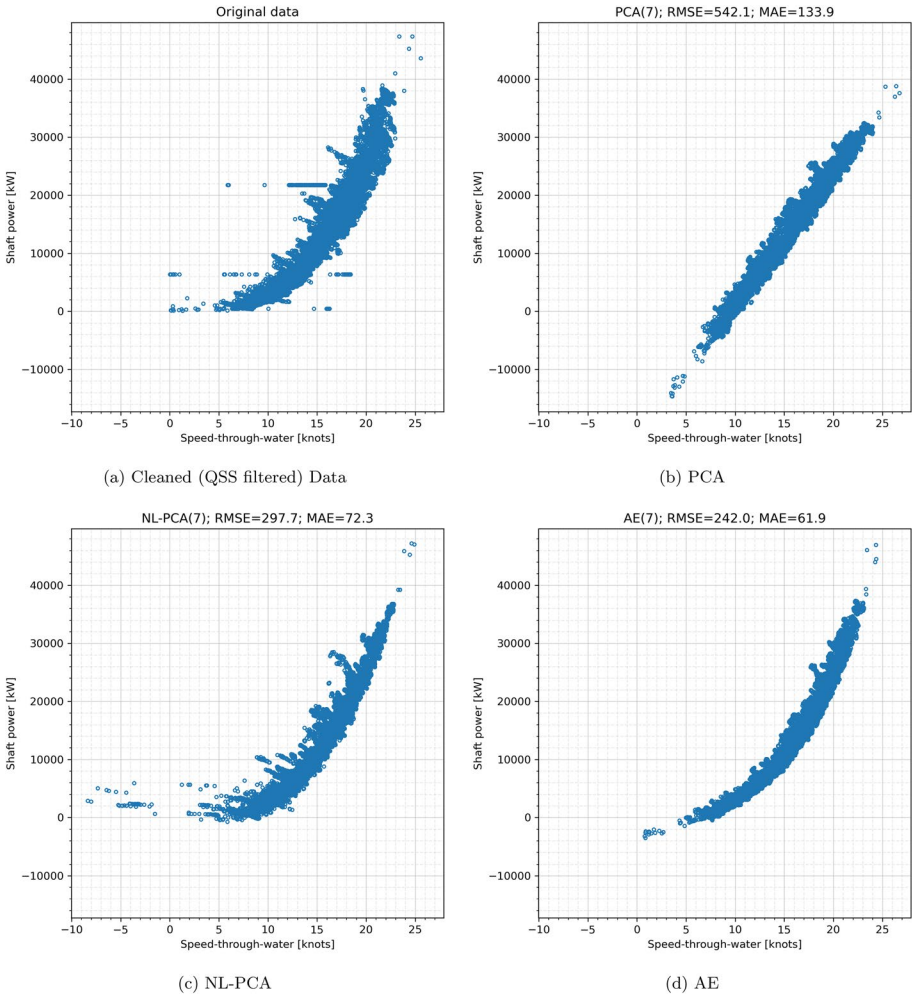
The physical meaning of each LV can be clearly understood by looking at the scaled correlation loadings (Fig. 3) and their actual numerical values (shown in Table 3). Besides, visualizing the projection of LVs in the real data space (as shown in Fig. 2) acts as a further confirmation. For instance, it is quite clear (from Figs. 2, 3 and Table 3) that the first PC from both the PCA models mainly represents the correlation between the speed, power, and rpm. In other words, the first PC represents the line along which the speed-though-water and shaft power would move if the rpm of the propeller is changed. A similar analysis can be done for the autoencoders model using its weight matrices, but it would be far more complicated due to the complex interconnections, especially in the case of more than 1 hidden layer.

Data reconstruction

In the case of PCA, the data is reconstructed by simply multiplying the eigenvectors (or PC loadings, $\mathbf{P}_A$) with the eigenvalues (or PC scores, $\mathbf{T}_A$) corresponding to each data sample, as given in eq. 2. Here, the number of prominent factors (or model dimensionality, A) is user-specified, which is generally based on the distribution of information or variance among the factors. For instance, in the case of the current dataset, the first 7 PCs contain almost 100% of the variance contained in the original dataset. Therefore, this dataset can be reconstructed using only the first 7 PCs. Although the reduction from 9 data variables to 7 PCs is not significant at all, the reconstruction process allows us to detect correlation-defying outliers, as shown in the next section ("Outlier detection").

In the case of autoencoders, the number of LVs is regulated by the number of neurons in the center-most hidden layer, and the data is reconstructed by using the trained network, after turning off the gradient calculations. Here, it is crucial to check if the reconstructed data depicts the same trends, linear and non-linear, as present in the original data. If not, then either the adopted hyperparameters or the number of LVs used to reconstruct the data should be reconsidered. However, including too many LVs for data reconstruction may result in relaying undesired noise in the reconstructed data (as the left-out LVs generally constitute uncorrelated noise in the data) and increase the computational time. Based on these considerations as well as an acceptably small reconstruction error, namely, Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE), 7 autoencoder LVs are used to reconstruct the current dataset. Figure 4 shows the reconstructed data obtained using all three models, i.e., PCA, NL-PCA, and autoencoders. The figure also shows the original data so that a visual comparison can be carried out. As expected, the reconstructed data from NL-PCA and autoencoders resembles the original data quite well, and the reconstruction error from both models is substantially small, whereas it is not the same for the linear PCA model. With only 7 LVs, the linear PCA model fails to capture the non-linear trends in the data.

The reconstructed data in Fig. 4 indicate that the linear PCA model is unable to capture the non-linear trends in the data. However, it is known that using the maximum number of PCs (same as the rank of the data matrix) would result in the dataset itself (with negligible reconstruction error) even in the case of the linear PCA model. Therefore, it may be interesting to investigate further and have a look at the last 2 discarded PCs. Figure 5 shows the data reconstructed using the linear PCA model with 7, 8 and 9 PCs. The data reconstructed using 8 and 9 (discarded) PCs is, in fact, able



(a) Cleaned (QSS filtered) Data (b) PCA

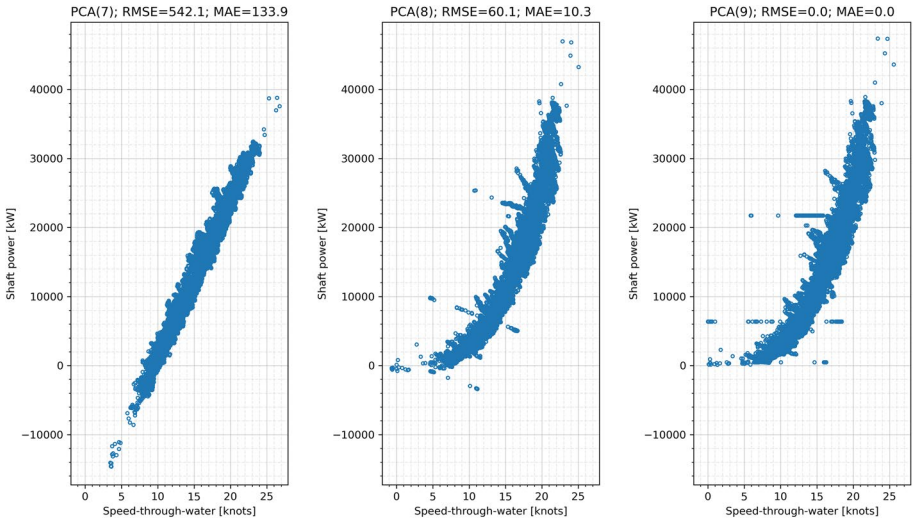
(c) NL-PCA (d) AE

Fig. 4 Data reconstruction with PCA, NL-PCA and autoencoders using 7 LVs. The original data (top-left) is also presented here for visual comparison. The reconstruction error (RMSE and MAE) is presented in the title of the corresponding subplots

to model the non-linear trends in the dataset, disproving the general belief that the linear PCA cannot model the non-linearities in the data. This, rather, indicates that the non-linear trends are extracted as separate PCs, which may be deprioritized by the model, i.e., obtained as one of the last PCs, resulting in being discarded when the data is factorized. Nevertheless, it may not be a good idea to use a linear PCA model for outlier detection in this case, as it models non-linearities using the last few PCs which may also be contaminated with undesired noise.

Outlier detection

The reconstructed data forms the basis for correlation-based outlier detection. Here, it is asserted that the data samples falling far away from the correlation trends, i.e., correlation-defying samples, are considered outliers. Based on this assertion,



(a) 7 PCs. (b) 8 PCs. (c) 9 PCs.

Fig. 5 Data reconstruction with linear PCA model using 7, 8, and 9 PCs. The reconstruction error (RMSE and MAE) is presented in the title of corresponding subplots

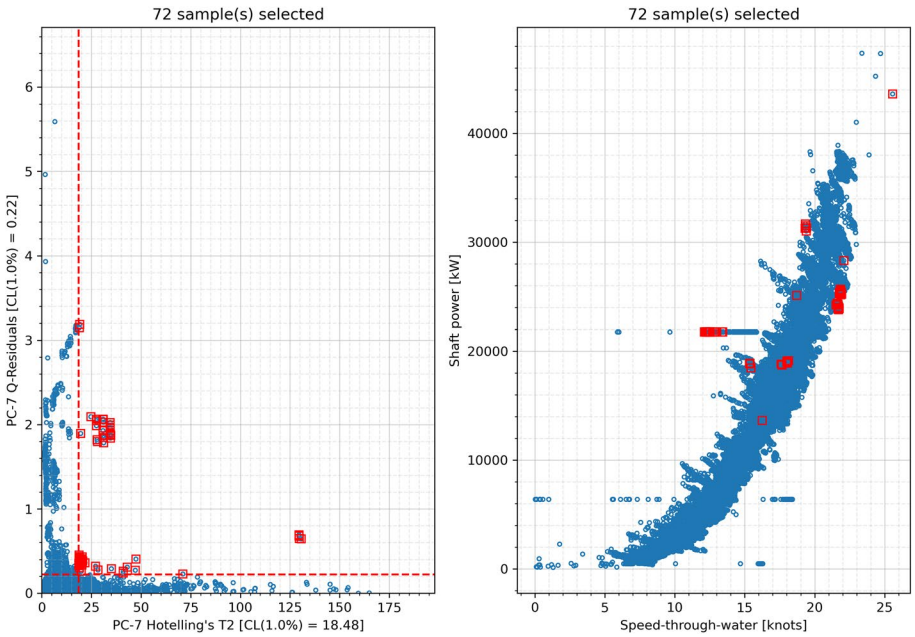


Fig. 6 Outliers detected (marked by red squares) with NL-PCA using 99% confidence limits on Q-residuals and Hotelling's T2 statistics

sample-wise reconstruction error is calculated, and the samples with high reconstruction error are recognized as outliers. However, in the case of PCA, the influence plots (explained in sect. "Principal component analysis (PCA)") present another way to detect outliers, based on the hypothesis that only samples with both high reconstruction error and high leverage should be considered outliers, as the samples with small leverage are basically harmless due to their small influence on the model. Figure 6

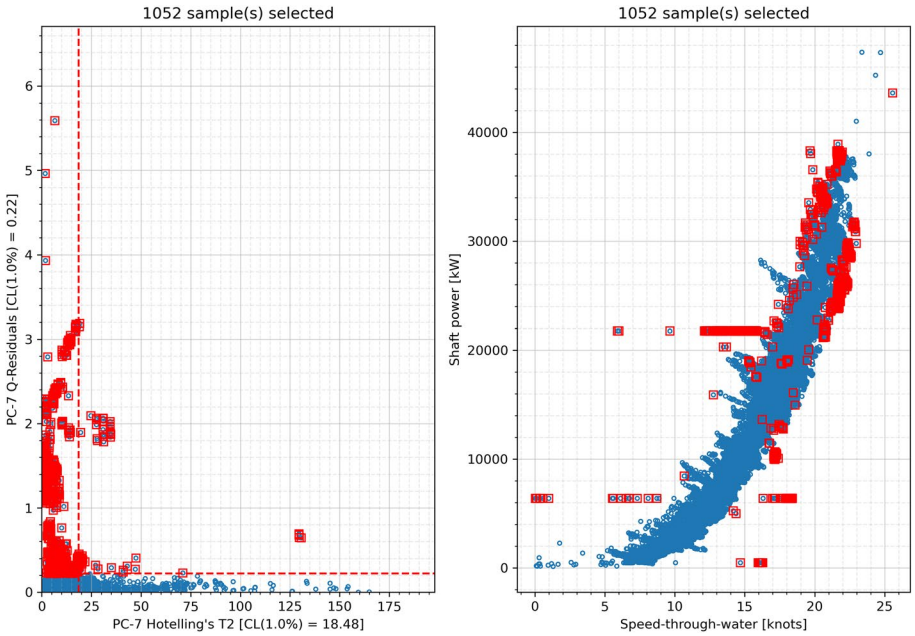


Fig. 7 Outliers detected (marked by red squares) with NL-PCA using 99% confidence limit only on Q-residuals

shows the outliers detected in the current dataset using the influence plot obtained from the NL-PCA model with 7 PCs. Here, the x-axis shows the leverage in terms of Hotelling's T^2 statistics [21], and the y-axis shows the reconstruction error in terms of Q-residuals (equivalent to normalized mean squared error) for each data sample. The samples above 99% confidence limits for Hotelling's T^2 statistics and Q-residuals are recognized as outliers.

The samples marked in Fig. 6 do not seem to include all the expected outliers (discussed in section "Data exploration & processing"), probably due to the exclusion of samples with small leverage (or Hotelling's T^2 statistics). Figure 7 shows the outliers detected with 99% confidence limit applied only on Q-residuals, i.e., also including samples with small leverage. Here, most of the expected outliers are marked, proving the effectiveness of influence plots and PCA for outlier detection. Moreover, the methodology avoids misclassifying rare but valuable samples in the lower speed-through-water range as outliers. In the case of autoencoders, it is not possible to calculate the leverage and statistical confidence limits due to its non-linear nature. However, it is possible to constitute an empirical distribution from the sample-wise mean squared error (MSE) and use it to detect potential outliers. Figure 8 shows the sample-wise MSE from the autoencoders model and the outliers detected with 99% confidence limit on the empirical distribution. In order to draw a fair comparison, similar results are presented from NL-PCA model in Fig. 9.

The results from Figs. 8 and 9 indicate that using the empirical distributions (based on sample-wise MSE) for outlier detection provides good results, especially in the case of autoencoders. However, the risk of using an empirical distribution should be kept in mind as it would always result in the removal of a fixed number of samples (depending on the chosen confidence limit), irrespective of how many outliers

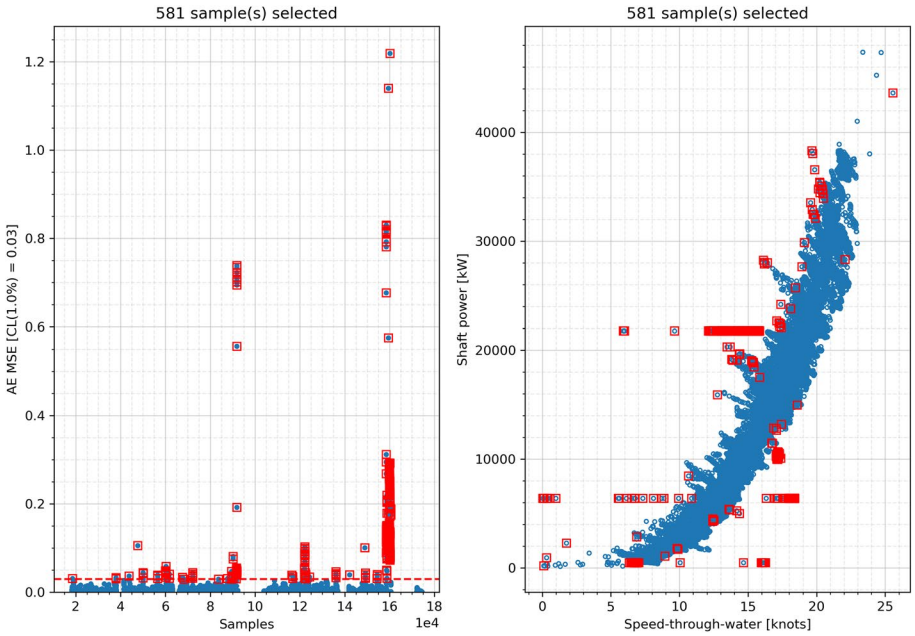


Fig. 8 Outliers detected (marked by red squares) with AE using 99% confidence limit on the empirical distribution of sample-wise MSE

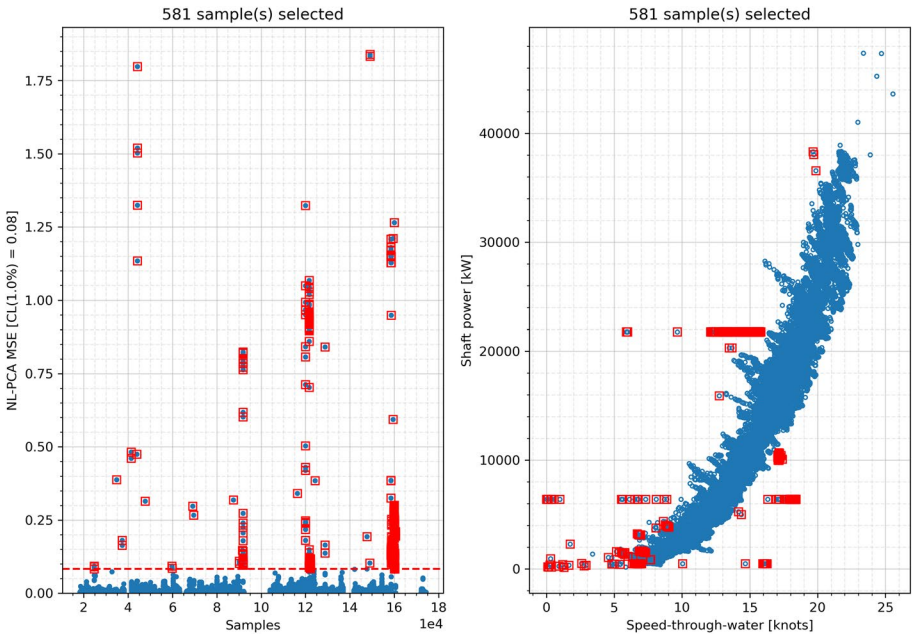


Fig. 9 Outliers detected (marked by red squares) with NL-PCA using 99% confidence limit on the empirical distribution of sample-wise MSE

are present in the dataset. If a known statistical distribution is used to get the confidence limits, the number of detected outliers would depend on the actual distribution of residuals, avoiding the problem of removing too many or too few samples. On another note, comparing the outliers detected from NL-PCA using the influence

plot and sample-wise MSE (i.e., Figs. 7 and 9), it is observed that the former results in detecting too many outliers in the higher range of speed and power. This is due to the cubic transformations applied on the speed-though-water and shaft rpm in NL-PCA model, resulting in higher numerical values of Q-residuals for samples with high speed and rpm. Thus, linearizing the residuals before carrying out outlier detection may be desirable, but the linearized residuals would not fit any known distribution, and therefore, no statistical distribution-based confidence limits can be obtained for them. However, it may be possible to use an ensemble of linear PCA models (without any non-linear transformation) in order to avoid this problem.

Conclusion

Outlier detection is a challenging task while handling a large amount of data. Most of the popularly-known outlier detection methods misidentify rare but valuable samples, observed in sparse regions of variable space, as outliers, thereby making outlier detection inefficient for unbalanced datasets. The current work mainly addresses this issue and constitutes the following:

- The proposed scheme, using PCA and autoencoders, detects correlation-defying outliers and avoids misidentifying rare but valuable samples as outliers in unbalanced datasets.
- The validation of the proposed scheme is carried out using in-service data recorded onboard a sea-going ship, which is highly unbalanced, and therefore, quite suitable for robust validation.
- The paper demonstrates that the linear but efficient PCA model, empowered by non-linear transformations (here referred to as NL-PCA) obtained using domain knowledge, is able to model non-linear correlations. The principal components (PCs) from NL-PCA model are also found best suited to understand the physical meaning of latent variables as well as study the correlations between the data variables.

As mentioned above, the validation for the proposed scheme is carried out using a dataset recorded onboard a ship over several sea voyages. Due to an unchanged propulsive state over large durations of its voyages, the onboard recorded datasets from ships are generally highly unbalanced, and they contain several rare but valuable samples, especially at low ship speeds. The proposed outlier detection scheme not only detects appropriate outliers but also avoids detecting rare samples as false positives. Therefore, it is proven effective for detecting outliers in unbalanced datasets, like ships' in-service datasets as well as any other similar datasets. Nevertheless, a thorough comparison with other outlier detection methods, discussed in sect. " Method selection", may be required. However, that may necessitate labeling outliers or obtaining a similar dataset with labeled outliers. Thus, it can be considered as a possible future work.

Acknowledgements
This study is part of the research project SFI Smart Maritime - Norwegian Centre for Improved Energy-Efficiency and Reduced Emissions from the Maritime Sector (RCN project number 237917).

Author contributions
Prateek Gupta: Conceptualization, methodology, software, formal analysis, investigation, data curation, visualization, writing—original draft. Adil Rasheed: methodology, supervision, writing—review and editing. Sverre Steen: resources, supervision, writing—review and editing.

Funding
Open access funding provided by Norwegian University of Science and Technology.

Availability of data and materials
The dataset used for the current study is strictly confidential as per the agreement with the data provider. Thus, it is neither possible to share it nor make it available publicly. Regarding code sharing, the code might be made public in the future on NTNU-IMT's GitHub page (https://github.com/NTNU-IMT), but it would not be considered part of this publication.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
Authors declare no competing of interests.

Received: 31 July 2023 Accepted: 18 May 2024
Published online: 13 June 2024

References

1. Edgeworth FY. XLI: on discordant observations. London Edinburgh Dublin Philos Mag J Sci. 1887;23(143):364–75. https://doi.org/10.1080/14786448708628471.
2. Chandola V, Banerjee A, Kumar V. Anomaly detection: a survey. ACM Comput Surv. 2009;41(3):1–58. https://doi.org/10.1145/1541880.1541882.
3. McClelland GH. Nasty data: unruly, ill-mannered observations can ruin your analysis. In: Reis HT, Judd CM, editors. Handbook of research methods in social and personality psychology. 2nd ed. New York: Cambridge University Press; 2014. p. 608–26.
4. Rousseeuw PJ, Leroy AM. Robust regression and outlier detection. Hoboken: John Wiley & Sons; 2005.
5. Gupta P, Kim Y-R, Steen S, Rasheed A. Streamlined semi-automatic data processing framework for ship performance analysis. Int J Nav Archit Ocean Eng. 2023;15: 100550.
6. Suboh S, Aziz IA. Anomaly detection with machine learning in the presence of extreme value—a review paper. In: 2020 IEEE Conference on Big Data and Analytics (ICBDA). 2020. p. 66–72.
7. Sakurada M, Yairi T. Anomaly Detection Using Autoencoders with Nonlinear Dimensionality Reduction. In: Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis—MLSDA'14. Gold Coast: ACM Press; 2014. p. 4–11. http://dl.acm.org/citation.cfm?doid=2689746.2689747. Accessed 22 July 2022.
8. Gupta P, Rasheed A, Steen S. Ship performance monitoring using machine-learning. Ocean Eng. 2022;254: 111094. https://doi.org/10.1016/j.oceaneng.2022.111094.
9. Zhao Y, Nasrullah Z, Li Z. Pyod: a python toolbox for scalable outlier detection. J Mach Learn Res. 2019;20(96):1–7.
10. Han S, Hu X, Huang H, Jiang M, Zhao Y. Adbench: anomaly detection benchmark. Adv Neural Inf Process Syst. 2022;35:32142–59.
11. Goldstein M, Dengel A. Histogram-based outlier score (HBOS): a fast unsupervised anomaly detection algorithm. In: KI-2012: Poster and Demo Track. 2012. p. 9.
12. Latecki LJ, Lazarevic A, Pokrajac D. Outlier detection with kernel density functions. In: International Workshop on Machine Learning and Data Mining in Pattern Recognition. 2007. p. 61–75.
13. Schölkopf B, Platt JC, Shawe-Taylor JC, Smola AJ, Williamson RC. Estimating the support of a high-dimensional distribution. Neural Comput. 2001;13(7):1443–71. https://doi.org/10.1162/089976601750264965.
14. Liu FT, Ting KM, Zhou Z-H. Isolation forest. In: 2008 Eighth IEEE International Conference on Data Mining. 2008. p. 413–22.
15. MacQueen J. Classification and analysis of multivariate observations. In: 5th Berkeley Symposium on Mathematical Statistics and Probability. 1967. p. 281–97.
16. Ester M, Kriegel H-P, Sander J, Xu X. A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. AAAI Press; 1996. p. 226–31.
17. Roberts S, Tarassenko L. A probabilistic resource allocating network for novelty detection. Neural Comput. 1994;6(2):270–84. https://doi.org/10.1162/neco.1994.6.2.270.
18. Breunig MM, Kriegel H-P, Ng RT, Sander J. LOF: identifying density-based local outliers. SIGMOD Rec. 2000;29(2):93–104. https://doi.org/10.1145/335191.335388.
19. Hawkins S, He H, Williams G, Baxter R. Outlier detection using replicator neural networks. In: International Conference on Data Warehousing and Knowledge Discovery. 2002. p. 170–80.

20. Jolliffe I. Principal component analysis. Springer. 2002. https://books.google.no/books?id=olByCrhjwIC.
21. Hotelling H. Analysis of a complex of statistical variables into principal components [Article]. J Educ Psychol. 1933;24(6):417–41. https://doi.org/10.1037/h0071325.
22. Jackson JE, Mudholkar GS. Control procedures for residuals associated with principal component analysis. Technometrics. 1979;21(3):341–9.
23. Chalapathy R, Menon AK, Chawla S. Robust, deep and inductive anomaly detection. In: Ceci M, Hollmén J, Todorovski L, Vens C, Džeroski S, editors. Machine learning and knowledge discovery in databases. Cham: Springer International Publishing; 2017. p. 36–51.
24. Gupta P, Steen S, Rasheed A. Big data analytics as a tool to monitor hydrodynamic performance of a ship. In: International Conference on Offshore Mechanics and Arctic Engineering. vol. 58844. 2019. p. V07AT06A059.
25. Golub G, Reinsch C. Singular value decomposition and least squares solutions. Numer Math. 1970;14(5):403–20. https://doi.org/10.1007/BF02163027.
26. Vandeginste B, Sielhorst C, Gerritsen M. NIPALS algorithm for the calculation of the principal components of a matrix. TrAC Trends Anal Chem. 1988;7(8):286–7. https://doi.org/10.1016/0165-9936(88)80007-4.
27. Jackson JE. A user's guide to principal components. Hoboken: John Wiley & Sons; 1991.
28. Thennadil SN, Dewar M, Herdsman C, Nordon A, Becker E. Automated weighted outlier detection technique for multivariate data. Control Eng Pract. 2018;70:40–9.
29. MacGregor JF, Kourti T. Statistical process control of multivariate processes. Control Eng Pract. 1995;3(3):403–14.
30. Müller K-R, Mika S, Rätsch G, Tsuda K, Schölkopf B. An introduction to kernel-based learning algorithms [Review]. IEEE Trans Neural Netw. 2001;12(2):181–201. https://doi.org/10.1109/72.914517.
31. Cadima J, Jolliffe IT. Loading and correlations in the interpretation of principle compenents. J Appl Stat. 1995;22(2):203–14. https://doi.org/10.1080/757584614.

Publisher's Note