Improving resilience of sensors in planetary exploration using data-driven models

MACHINE
LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
12 May 2023

REVISED
17 July 2023

ACCEPTED FOR PUBLICATION
10 August 2023

PUBLISHED
4 September 2023

PAPER

Improving resilience of sensors in planetary exploration using data-driven models

Dileep Kumar[1], Manuel Dominguez-Pumar[1,*], Elisa Sayrol-Clols[2], Josefina Torres[3], Mercedes Marín[3], Javier Gómez-Elvira[3], Luis Mora[3], Sara Navarro[3] and Jose Rodríguez-Manfredi[3,*]

1 Universitat Politècnica de Catalunya (UPC), Barcelona, Spain
2 TecnoCampus, Universitat Pompeu Fabra (UPF), Mataró, Spain
3 Centro de Astrobiología (INTA-CSIC), Madrid, Spain
* Authors to whom any correspondence should be addressed.

E-mail: manuel.dominguez@upc.edu and manfredi@cab.inta-csic.es

Keywords: space sensor systems, wind sensor, machine learning, deep learning, soft sensor

Abstract

Improving the resilience of sensor systems in space exploration is a key objective since the environmental conditions to which they are exposed are very harsh. For example, it is known that the presence of flying debris and Dust Devils on the Martian surface can partially damage sensors present in rovers/landers. The objective of this work is to show how data-driven methods can improve sensor resilience, particularly in the case of complex sensors, with multiple intermediate variables, feeding an inverse algorithm (IA) based on calibration data. The method considers three phases: an initial phase in which the sensor is calibrated in the laboratory and an IA is designed; a second phase, in which the sensor is placed at its intended location and sensor data is used to train data-driven model; and a third phase, once the model has been trained and partial damage is detected, in which the data-driven algorithm is reducing errors. The proposed method is tested with the intermediate data of the wind sensor of the TWINS instrument (NASA InSight mission), consisting of two booms placed on the deck of the lander, and three boards per boom. Wind speed and angle are recovered from the intermediate variables provided by the sensor and predicted by the proposed method. A comparative analysis of various data-driven methods including machine learning and deep learning (DL) methods is carried out for the proposed research. It is shown that even a simple method such as k-nearest neighbor is capable of successfully recovering missing data of a board compared to complex DL models. Depending on the selected missing board, errors are reduced by a factor between 2.43 and 4.78, for horizontal velocity; and by a factor between 1.74 and 4.71, for angle, compared with the situation of using only the two remaining boards.

1. Introduction

Sensor reliability is one of the key properties of sensors and sensor networks. This is particularly true in the case of space exploration, since the effect of missing data can strongly affect science output, and at the same time, replacing or repairing a damaged sensor is usually not possible. The case example studied in this paper will be the wind sensor of the TWINS instrument, of NASA InSight mission, launched in 2018 [1, 2]. Even though this wind sensor has not suffered any damage on Mars surface, other wind sensors have suffered partial damage. This is the case, for example, of the MEDA wind sensor [3], which has been partially damaged on Mars due to the presence of flying debris generated by Dust Devils [4]. Therefore, the use of techniques to improve sensor resilience in space exploration (and other fields) is a key research objective.

Atmospheric monitoring on Mars plays an important role in developing scientific knowledge of the planet. It involves various sensors to characterize the environmental dynamics (air temperature, pressure, sky opacity, wind velocity, etc). Among the various sensors used, wind sensors have been considered as key elements for the exploration of the atmosphere [5, 6]. Thus, it is crucial to receive data from sensors using a reliable approach. This implies uninterrupted monitoring of wind activity using the received data.

With the increased use of sensors, sensor problems have also been common in different domains such as industrial condition monitoring systems, traffic monitoring systems, marine systems, and healthcare systems [7–9]. The possible reasons behind data discontinuities, or stoppages, can be network outages, power issues, or sensor failures. Particularly, sensors used in space exploration missions operate under harsh environmental conditions [10]. Considering the importance of sensors in space exploration, they must be resilient against such scenarios, which can negatively affect the missions.

Therefore, various methods have been applied to recover the data of a sensor in case of long discontinuities. Data-driven methods come into play to provide data estimations for unavailable sensor using data from other sensors. The estimations of dependent variables using independent variables is known as regression problem [11]. Various data-driven models have been used to develop soft or virtual sensors in different applications [12–15]. Among these data driven models, machine learning (ML) and deep learning (DL) models have emerged as promising solutions for implementing soft sensors or predicting time series data. These models allow solving regression problems through training processes, which maps inputs to outputs. The data-driven methods provide reliable estimations through modeling of other process variables. However, it becomes difficult for data-driven approaches to accurately model data due to complex characteristics of available data for estimations [12]. Fortunately, ML and DL methods have been surpassing statistical, signal processing methods investigated for complex data analysis. Moreover, DL methods can learn rich features from non-linear data owing to their multilayered and hierarchical structure. The key features of DL models such as high accuracy, self-feature learning and the capability of working with raw/noisy data, have been capturing the attention of data scientists or researchers [16–18].

In this paper, a comparative analysis of various ML and DL methods is carried out for making complex sensors resilient against long data discontinuities. The comparative analysis among various data-driven models such as polynomial regression (PR), k-nearest neighbors (KNNs), random forest (RF), extreme gradient boosting (XGboost), multi layer perceptron (MLP), long short-term memory (LSTM), and bidirectional LSTM (Bi-LSTM) is conducted for implementing a soft wind sensing board. These ML and DL algorithms have been widely explored for similar problems with different model configurations. In fact, the performance of these algorithms not only rely on their hyperparameters but also depends on the nature of data. Depending on data complexity, sometimes a simple model can yield better results than a complex one [19, 20]. Thus, all aforementioned models have been experimented for comparative analysis in this work.

Various data-driven methods have been used for sensor hysteresis compensation [21], sensor variable estimations and localization [22, 23], sensor output classification [24], sensor calibration [13], and sensor fault diagnosis [25]. Considering the missing data imputation problem, there have been multiple studies conducted in different domains using data-driven approaches for imputing multivariate or univariate missing data using relevant input data features. A recent study [26] investigated multiple methods for imputing missing environmental data. The methods including missForest (MF), multivariate imputation by chained equations (MICEs), and KNNs were employed to address the random missing data problem. To conduct the proposed study, the authors artificially induced missing data in the environmental databases with varying percentages. The experiments were conducted on ten different pre-processed datasets and the achieved results demonstrated that the MF model was able to outperform the MICE and KNN models. The MF model allowed to reduce the missing data imputation errors significantly and KNN demonstrated less processing time compared to the other two models. Moreover, the researchers conducted a real case study on the performance monitoring station of Quebec waste water plant which showed the effectiveness of the proposed method in terms of addressing missing data problem in diverse environmental databases. Another relevant investigation [27] carried out experimentations on five data imputation methods with a US Credit Bureau dataset. The dataset after pre-processing included 329, 917 data samples and 153 variables and different percentages of data were removed to establish the datasets for the proposed methods. The methods included Amelia, MICE, mi, Hmisc, and MF. The performance of these methods was evaluated using root mean squared error (RMSE) against the conventional mean based imputation method. Based on the achieved results, the authors concluded that MF yielded the best performance and mi model yielded the worst performance. However, there was no impact on choice of an imputation model with change in data missing percentage. More recently, the authors of [28] have put efforts on experimenting potential of diffusion models on missing data imputation for both the categorical and numerical variables. The proposed method was applied to seven different open-source benchmarking datasets. The achieved results on the benchmark dataset demonstrated the effectiveness of the conditional score-based diffusion models for tabular data (TabCSDI) model compared to the well-known existing imputation methods including Mean, MICE (linear), and MF. Furthermore, the model provided evident better performance in imputing categorical variables with feature tokenization. For further improvement through the diffusion models, the authors pointed to a focus on improving inference time, model architecture improvement, and loss function optimization.

Compared to the existing work in the domain of sensors and missing data imputation, we present a data-driven approach to improve resilience of the sensor under circumstances of continuous missing variables of a complex wind sensor in space (Mars). This study focuses on imputing the missing data variables through feature learning from the data variable of remaining sensing units of the sensor. The investigation targets imputation of multiple numerical variables. Moreover, this work is an extension of our partial preliminary work [29, 30], and we focus on a large number of different models, and we improve the results obtained previously.

The proposed methods are tested with the intermediate variables of the wind sensor of the TWINS instrument, of the InSight mission. The key contributions are:

(1) We propose a soft wind sensing board based on a data-driven model for recovering the missing intermediate data of wind sensor feeding an inverse algorithm (IA) from which the final sensor output is established. In the case of missing data for longer periods from a board, the use of data predicted by a data-driven model can assist in improving sensor accuracy by recovering the intermediate variables.
(2) For implementation of the soft wind sensing board, a comparative analysis of multiple data-driven methods has been carried out. The analysis of different ML and DL models allows to select simple and efficient solution for the soft sensor.
(3) We employ power transformation for data pre-processing. The pre-processing method makes the dataset more like a Gaussian distribution, which allows the models to perform smoothly and learn better representation from the data.
(4) The proposed methods allow predicting multivariate data for one wind sensing board using the data of the two boards of the wind sensor of the same boom. The comparative analysis of various data-driven methods reveals that even a simple method like KNN works effectively for the proposed soft sensor owing to the nature of the data and model architecture.

The paper is organized as follows: section 2 reports the wind sensor and its working principle. Section 3 describes the dataset used in the research. Section 4 describes the algorithms, section 5 discusses experimental configuration and evaluation metrics. Section 6 reports and discusses the results achieved in this research. Finally, section 7 concludes the paper.

2. Mars wind sensor

The wind sensor of the TWINS instrument is very similar to the one in the REMS instrument of the Mars science laboratory (MSL) mission, launched in 2011 [16, 17]. Both sensors are based on hot film anemometry implemented on two booms. At the surface of each of the booms, there are three boards in which the local wind velocity, at three points is monitored. Local 3D wind recovery is obtained at each boom using the data from its three boards. Finally, a high level algorithm implements the final wind selection from the two local wind estimations (one per each boom), based on the optimality of the wind angle incidence on the booms. Figure 1 shows the placement of the booms on the lander deck.

Figure 2 depicts a detail of one of the booms, and one of its boards can be clearly seen. All boards in each boom are oriented sideways at an angle of 120° between them. The booms are installed on the lander's deck in such a way that their horizontal plane is parallel to the ground and are placed diametrically opposite to each other. The booms are placed in this position to minimize the impact of other components on the InSight's deck. This also ensures that at least one of the booms is toward wind flow at all the times [2].

The detail of each board has been presented in [2, 31, 32]. The objective is to monitor convection heat transfer from four silicon dice which are heated above the temperature of the air and are kept at a constant temperature difference with regard to a fifth die, a cold die, which is not heated, on the same board. All dice have been placed on top of four supporting rods in order to thermally isolate them from the boom. Figure 3 shows a picture of one of the prototypes of the boards.

The hot dice have three kinds of Pt resistors deposited: one resistor is used for heating the sensor, a second one is used to measure the temperature (sensing resistor) and the last one is used as reference of the overheat for the measurements (it helps to set the temperature difference between the hot and cold dice). The cold die at the extreme of the printed circuit board uses only the sensing resistor to measure the temperature. As it has been mentioned before, the board operates under constant temperature difference between the hot dice and the cold die. Changes in the wind velocity observed by the boards are compensated by the closed loops controlling the temperatures of the hot dice [31]. The control loops implement thermal sigma-delta modulators.

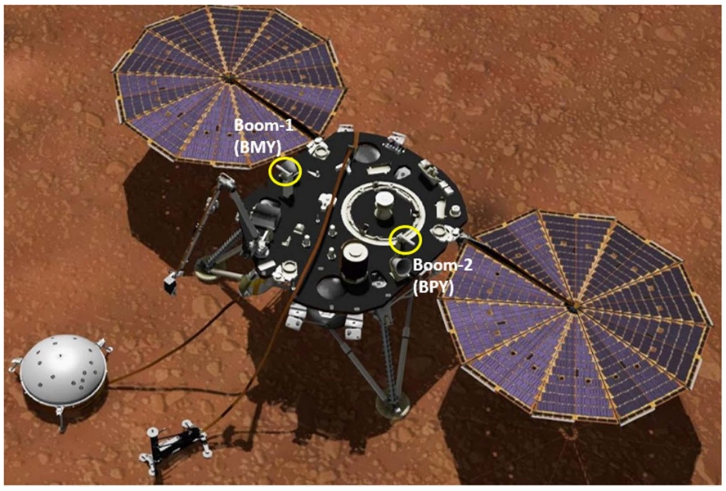


Figure 1. InSight lander on surface of the Mars.

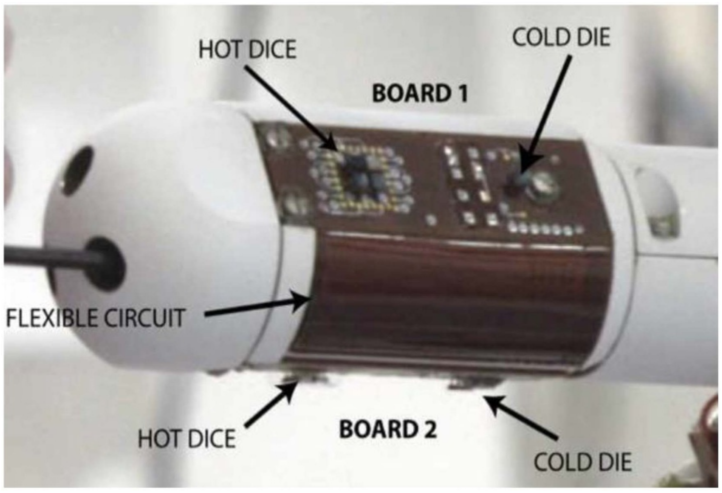


Figure 2. Wind sensing boom arrayed with the sensing boards.

The wind sensor data used in this paper is acquired at the sampling rate of 1 Hz [1]. For a better interpretation of the sensor's behavior over time, the variables from the four dices are reduced to two variables obtained as difference between the normalized convective conductance of the thermal sigma-delta modulators as follows [31]:

$$B_{\text{LONG}} = \text{North} - \text{South} = (A + D) - (B + C) \tag{1}$$

$$B_{\text{TRANS}} = \text{East} - \text{West} = (A + B) - (C + D)\,. \tag{2}$$

2.1. Generation of the intermediate variables

Calculating the intermediate variables B_LONG and B_TRANS requires first to obtain the convective power exchanged with the atmosphere by each of the four dice of the transducer board. The injected power (P_{inj}) for each die can be directly calculated using the sigma-delta pulses readings [31], to then calculate the convective power (P_{conv}). The conductive (P_{cond}) and radiative (P_{rad}) powers are calculated using the coefficients calculated in the thermal vacuum dice characterization tests [2]. The convective conductance (G) can be simply obtained dividing the convective power by the overheat as given in equation (4), which is directly related with the convective coefficient h,

$$P_{\text{conv}} = P_{\text{inj}} - P_{\text{cond}} - P_{\text{rad}} \tag{3}$$

$$G = \frac{P_{\text{conv}}}{(T_{\text{hot}} - T_{\text{air}})}\,. \tag{4}$$

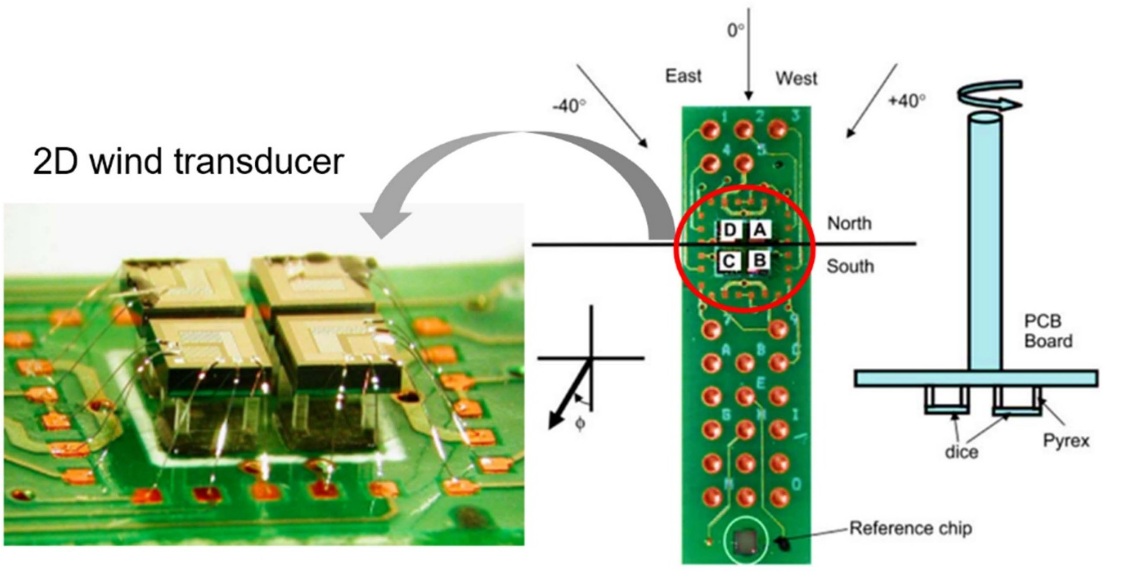


Figure 3. Wind sensing boom arrayed with the sensing boards.

In order to make the convective conductance non-dependent on temperature, pressure and the conductivity of the fluid, the estimator B, the normalized convective conductance, is obtained, and using equations (1) and (2) B_LONG and B_TRANS can be finally obtained,

$$B = \frac{GL}{KPr^{1/3}} \tag{5}$$

where L is the characteristic die dimension, K is the fluid thermal conductivity, and Pr is the Prandtl number.

2.2. Inverse algorithm (IA)

The calibration of the sensor in the wind tunnel as described in [2] produces a calibration file relating Reynolds number (used as wind speed estimator), wind direction and the B_LONG–B_TRANS variables for each board (1–3). This calibration file has been interpolated for a fine Reynolds and direction mesh so that a lookup algorithm outputs the most similar Reynolds-direction for a given reading of B_LONG$_{1-3}$ and B_TRANS$_{1-3}$.

It must be noted that the IA computes the local wind velocity and angle at each boom, independently of the other, using the boards of each boom. The local wind velocities at each boom can be different, owing to the possible obstacles on the lander deck in the direction of the incoming wind. Additionally, for each boom, wind retrieval is optimal for a certain range of horizontal angles. For example, when the wind is coming from the rear of a boom, the wind inference is not correct, because the rear part of the boom is generating an aerodynamical distortion of the local wind pattern. It has been determined [1] that for all angles, at least one boom is always in a good position, and therefore, the last part of the IA includes a routine determining which local boom velocity is the one finally considered as correct (and placed in the Planetary Data System, PDS).

2.3. Restricted inverse algorithm (RIA)

The first possibility given a failure of a board is to use a restricted version of the IA, RIA, in which the lookup algorithm uses only a database populated with the Reynolds-wind direction associated to the two remaining pairs of B_LONG/B_TRANS (thus the missing B_LONG/B_TRANS pair is not used), as illustrated in figure 4. Other than the missing B_LONG/B_TRANS entries, this restricted algorithm is identical to the regular IA.

As mentioned before, the objective of this paper is to improve the resilience of complex sensors that use multiple intermediate data for generating the output signals. The proposed algorithm will be tested with data from the wind sensor of the TWINS instrument. We will consider scenarios with missing data from one board of any of the two booms of the sensor. The ML or DL network will be used to infer the B_LONG and B_TRANS intermediate variables of the missing boards, so that all the six intermediate variables can be fed into the IA. A comparison will be made with using only the RIA and the data from the two remaining boards per boom.

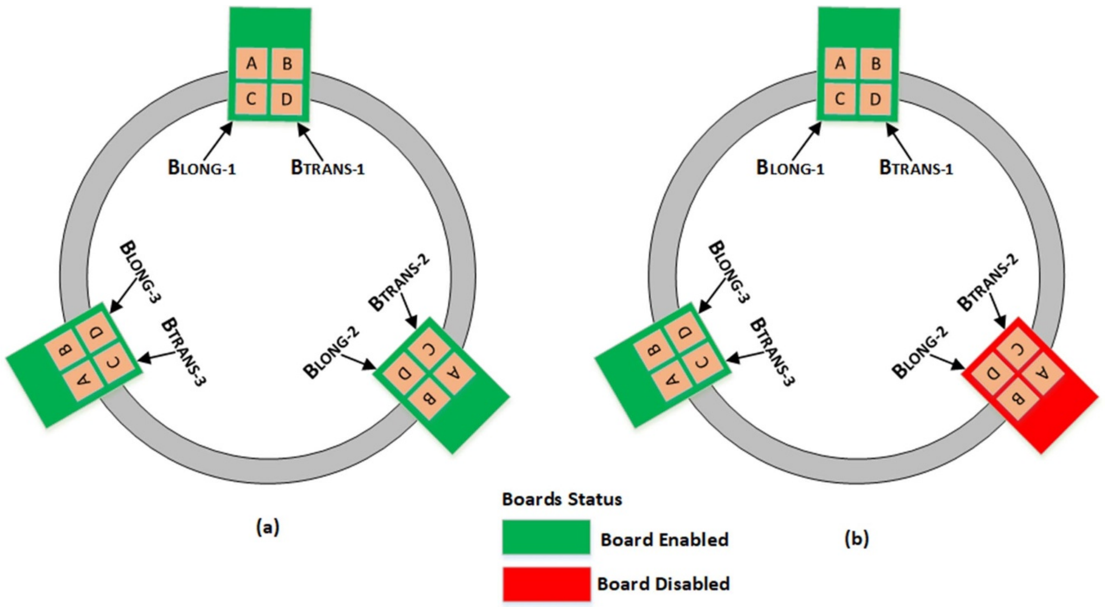


Figure 4. Wind sensing boom with (a) all working boards, and (b) one board disabled. The RIA uses the data from the remaining boards, whereas in this work we propose to obtain a prediction of the missing data variables obtained while the sensor is in Mars and fully operational.

Here, we show that the built-in correlations in the sensor can be further exploited because we use it for training, the data obtained while the sensor is on Mars surface. In the development of most missions, all this data will be much more than the data obtained during the calibration process. The amount of information used for training, validation and test is therefore much larger than the original set used for calibration.

3. Dataset and pre-processing

The TWINS instrument dataset, provided by the Centro de Astrobiologia (CAB, INTA-CSIC), is used for the proposed work. The dataset includes multiple comma-separated values files and each file contains the data for one sol (one day of Mars). The files include time-series data of the two booms (BPY and BMY), temperature, pressure, and wind sensors installed on the InSight lander. Each boom contains three boards. Thus, the variables to be used for the proposed models are BMY_LONG$_{1-3}$, BMY_TRANS$_{1-3}$, BPY_LONG$_{1-3}$, and BPY_TRANS$_{1-3}$. Where the B_LONG and B_TRANS are the orthogonal parameters of normalized convective conductance obtained using equations (1) and (2).

To investigate the proposed method, the individual data files of TWINS instrument from sol 126 to sol 254 (one sol = one Martian day) are loaded and combined. The combined dataset of 128 sols is then cleaned for removing the rows with no values. In the next step, the combined dataset is split into three sets with the percentage of 70%, 20%, and 10% as training set, validation set, and testing/cross-validation (CV) set, respectively. The training and validation sets comprise of 3534 901 data samples. The testing or CV set includes 354 491 data samples. After splitting the dataset into respective sets, it is necessary to scale the data for efficient representation learning. Sometimes, datasets need to be transformed into a Gaussian distribution to achieve optimal performance from a complex dataset. For the proposed work, we apply a power transform to make the data distribution more Gaussian-like. Moreover, the transformation has applied to all the used data-driven methods except PR.

The Yeo–Johnson (YJ) transformation is an upgraded form of the Box-Cox transformation. The transformation is applied on the InSight mission dataset to transform it into a Gaussian-like distribution, which allows DL model to effectively learn features from the non-linear data. It stabilizes the variance and normalizes the data distribution. The YJ transformation works for both positive and negative data variables. It makes data distribution symmetric about mean value for better analysis [33]. It has been extensively used for pre-processing datasets to improve the data analysis accuracy [34, 35]. The mathematical function for the YJ power transformation is expressed as follows:

$$P_d = H(q,\delta) = \begin{cases} \frac{(q+1)^{\delta}-1}{\delta} & ; q \geqslant 0, \delta \neq 0 \\ \log(q+1) & ; q \geqslant 0, \delta = 0 \\ -\frac{(-q+1)^{2-\delta}-1}{2-\delta} & ; q < 0, \delta \neq 2 \\ -\log(-q+1) & ; q < 0, \delta = 2 \end{cases} \tag{6}$$

where P_d represents the transformed data, q is the input data, and δ is a transformation parameter. The transformation parameter is any real number between 0 and 2. The parameter is optimized by the transformation method to transform a data distribution closer to a normal distribution.

We apply the YJ transformation to the dataset to improve the data quality. The pre-processing allows the DL model to efficiently learn representations from a Gaussian-like data distribution. The predicted data is transformed back to the original scale by applying the inverse YJ transformation.

4. Soft wind sensing board based on ML and DL models

Several ML and DL algorithms have been used, namely PR, KNN, RF, XGboost, MLP, LSTM, and Bi-LSTM models. This section provides a short theoretical description of these models implemented for multivariate data predictions.

Each of these models is explained in the following subsections:

4.1. Polynomial regression (PR)

A PR model allows to learn non-linear relationships between dependent and independent variables computing polynomials features. Multivariate PR can be represented by equation (7) [36]:

$$y = w_0 + \sum_{l_1=1}^{m} w_{l_1} x_{l_1} + \sum_{l_1=1}^{m}\sum_{l_2=l_1}^{m} w_{l_1 l_2} x_{l_1} x_{l_2} + \ldots + \sum_{l_1=1}^{m}\sum_{l_2=l_1}^{m} \cdots \sum_{l_k=l_{k-1}}^{m} w_{l_{1 l_2} \ldots l_k} x_{l_1} x_{l_2} \ldots . x_{l_k} \tag{7}$$

where k is polynomial order, $w_{l_1}, w_{l_1 l_2} \ldots . w_{l_1 l_2 \ldots .. l_k}$ represent PR coefficients, and $x_{l_1}, x_{l_2} \ldots . x_{l_k}$ represent input variables. We are using the least squares method to find the polynomial coefficients through minimizing sum of squared error of the predicted outcomes to the ground-truth values. In this research, we use PR method to predict TWINS wind sensor data. Furthermore, a grid-search approach, which is explained in section 5 is utilized to select the optimal number of degrees for the analysis.

4.2. K-nearest neighbor (KNN)

KNN is a widely used method for regression and classification problems owing to its simple learning process. For this algorithm, selection of an optimal k value is of prime importance to achieve effective performance [37]. It uses feature similarity learned from nearest values of training data to predict new data instances. It employs a few simple steps to predict a new value from its functional subspace. First, it computes the distance between input data samples and available data samples and selects KNNs of input data samples. Then, yields output predictions as an average value of the selected KNNs. Thus, performance of KNN methods depends on the optimal selection of the k-value. KNN is a simple and efficient learning model. However, it becomes a lazy learning model with large datasets because it performs lots of distance computations with a large training data size [38]. We employ the KNN model for multivariate regression with Minkowski distance metric, which is given by equation (8):

$$d = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p} \tag{8}$$

where x_i is query data sample and y_i is available data sample. The p is selected to use it as different distance metrics like if p is 1 then it is used as Manhattan distance. If p value is 2 then it becomes Euclidean distance. In this research, the p value is obtained through the grid-search approach.

4.3. Random forest (RF)

Ensemble learning enables for improving model performance through using multiple learners together. The RF algorithm was created by Breimen [39] with the idea of achieving better performance through ensemble learning. The RF model is a type of ensemble learning which consists of multiple decision trees which allows reduced overfitting and improved generalization performance through solving the problem of large variance by averaging several deep decision trees. This is also a supervised learning method which is used for both classification and regression problems. It provides predictions by utilizing multiple unrelated decision trees with a set depth. The final predictions are obtained through average of all decision trees in an RF model. This model is computationally efficient and easy to implement owing to its fewer hyperparameters such as tree size, tree depth, and number of variables. Moreover, there have been many examples in which the RF model is employed for a soft-sensor implementation [40, 41]. In this research, RF model is used for multivariate regression problem.

4.4. Extreme gradient boosting (XGBoost)

XGBoost is an upgraded form of gradient boosting decision method [42] which was introduced by Chen and He [43]. XGboost has the capability to efficiently construct boosted trees that can operate in parallel and solve regression and classification problems. For achieving higher prediction performance, XGboost continuously adds and grows depth of trees by feature splitting. Once the model is trained, k trees are obtained with scores corresponding to each tree. The final prediction is obtained from scores of k trees.

It tries to optimize an objective function with a gradient boosting framework and a regularization term to avoid the overfitting problem. Compared to general gradient boosting, XGBoost has highly scaled and flexible model structure that improves its speed and makes it less computationally intensive [44]. We use XGBoost model to predict multivariate data of the wind sensing board of TWINS instrument.

4.5. Multi layer perceptron (MLP)

MLP is a non-linear and widely used DL topology that consists of at least three layers including input layer, hidden layer, and output layer. These layers and interconnected by basic units called neurons. Within a MLP network, three main operations are carried out. First, input vectors are multiplied by weight matrices. In the second step, a bias value is added to the weighted input matrices. Lastly, an activation function is applied and the output is obtained as a weighted sum of the inputs. The activation function allows to ensure that each input data sample is mapped to different space in the final output vector [45, 46].

The MLP network training part consists of two processes including forward propagation and backpropagation. Forward propagation computes output and backpropagation computes error and update the weights based on the error. Backpropagation updates weights using differentiation and the chain rule. The MLP network can include multiple hidden layers which learn the model parameters [47]. The learned parameters are further fed to the output layer which yields outputs with linear activations, in case of regression problems. In this work, MLP model is implemented for multivariate regression problem.

4.6. Long short-term memory (LSTM)

LSTM models have been extensively used owing to their memory-based architecture. Moreover, they have been able to provide promising results in continuous data predictions by avoiding the gradient vanishing problem [48], which typically represents a problem for deep neural networks.

The LSTM topology was proposed by Hochreiter and Schmidhuber [48], to overcome the limitations of recursive neural networks. The capabilities of LSTM, such as memorizing past data sequences and discarding unnecessary information using its gated architecture make it a good choice for problems with sequential data. Each LSTM unit consists of four sub-units or gates as shown in figure 5, which includes three gates and one memory unit. The gates are shared among all the cells in the unit. Each LSTM unit receives three parameters as input (x_t), and state of previous cell (C_{t-1}), and previous hidden state (h_{t-1}). After, processing of the current input information by the three gates, the gates states are normalized by a tanh activation function. The gates regulate the flow of information within LSTM unit and yield outputs.

The operation of an LSTM unit is defined as follows:

Memory cell (C_t) can be considered as a mini-state machine that stores or memorizes the state of the unit.

Forget gate (f_t) decides what percentage of the information from the previous memory cell needs to be discarded or preserved. It decides through combining output of previous cell (h_{t-1}) and current input (x_t) and applying a sigmoid activation function (σ) as expressed in equation (10). The sigmoid function is expressed in equation (9), and yields a value between 0 and 1. Where, 0 means forgetting and 1 means memorizing the state. In the subsequent step, it is multiplied with the previous cell state (C_{t-1}) as given in equation (11),

$$\sigma(t) = \frac{1}{1+e^{-t}} \tag{9}$$

$$f_t = \sigma\left(w_f \cdot [h_{t-1}, x_t] + b_f\right). \tag{10}$$

Input gate (i_t) selects the input value that needs to update the current memory state. It selects through combining output of previous cell (h_{t-1}) and current input (x_t) and applying sigmoid activation function (σ) as given in equation (12). Then, tangent hyperbolic function (tanh) is applied on similar combination of parameters to generate candidate cell state vector ($\tilde{C}_t$). The tanh function given in equation (11), provides an

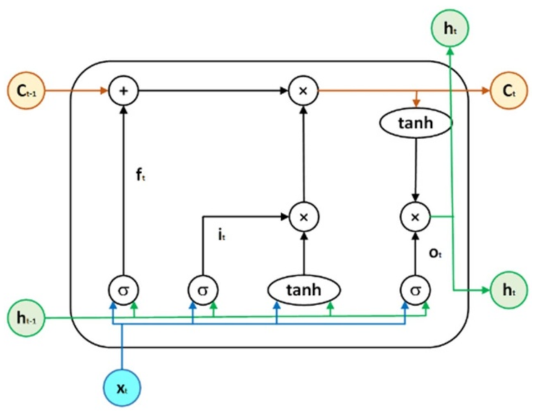


Figure 5. A LSTM unit.

output between range $(-1, 1)$. Subsequently, the current cell state (C_t) is obtained from the addition of forget gate f_t and input gate i_t outcomes after element wise multiplication with previous and new cell state, respectively. The new cell state (C_t) is expressed as follows:

$$\tanh(t) = \frac{e^t - e^{-t}}{e^t + e^{-t}} \tag{11}$$

$$i_t = \sigma\left(w_i \cdot [h_{t-1}, x_t] + b_i\right) \tag{12}$$

$$\tilde{C}_t = \tanh\left(w_c \cdot [h_{t-1}, x_t] + b_c\right) \tag{13}$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t. \tag{14}$$

Output gate (o_t) produces the output at time 't' based on the application of the sigmoid function on the current input and the state of the previous hidden layer (h_{t-1}). The new hidden (h_t) is produced by element wise multiplication of the output gate and new the cell state of the cell (C_t) which is activated by function tanh,

$$o_t = \sigma\left(w_o \cdot [h_{t-1}, x_t] + b_o\right) \tag{15}$$

$$h_t = o_t{}^* \tanh(C_t). \tag{16}$$

$\tilde{C}_t$ is the candidate cell computed using equation (13) which stores the updated information. The variables w_i, w_c, w_f, and w_o are the weight matrices for the gates and cell state. Similarly, b_i, b_c, b_f, and b_o are the bias values for the gates and cell state. The w_s and the b_s are the ones that are estimated during the training of the model. Lastly, * symbol represents element wise operations.

In this work, stateless LSTM layers are used in which LSTM cells reinitializes cell states for every new batch of data during the training of the model in order to reduce complexity of the network and have a faster training process.

4.7. Bidirectional LSTM (Bi-LSTM)

Bi-LSTM is an advanced variant of LSTM algorithm, it computes the final output of the hidden layer by combining the forward layer and backward layer [49] as shown in figure 6. Where, $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ represent hidden vector at time t of LSTM forward layer and backward layer, respectively. As depicted in figure, both the $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ parameters are only related to their respective LSTM layers and are independent of each other. Firstly, it computes the parameters in forward direction from time 1 to t. Then, the hidden layer parameters are updated each time. Secondly, it computes the parameters in backward direction from time t to 1. Lastly, the final output of Bi-LSTM (y_t) is obtained by combining both the results of forward and backward computations by the weighted connection of these two hidden layers in the figure 6. The forward and

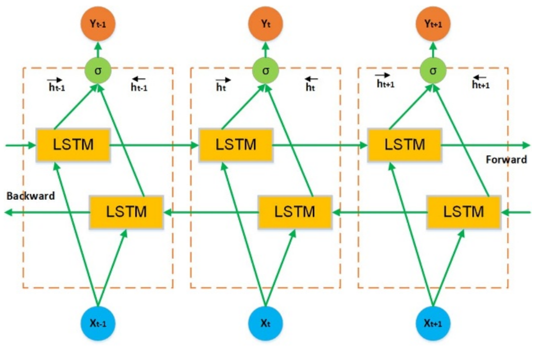


Figure 6. Bi-LSTM structure.

backward computations use similar operation as that of standard LSTM layer which is described by following expressions:

$$\overrightarrow{h_t} = \text{LSTM}\left(x_t, \overrightarrow{h_{t-1}}\right) \tag{17}$$

$$\overleftarrow{h_t} = \text{LSTM}\left(x_t, \overleftarrow{h_{t+1}}\right) \tag{18}$$

where LSTM (•) denotes LSTM network.

The final output vector of Bi-LSTM layers is obtained using the expression as follows:

$$y_t = \sigma\left(W_{\overrightarrow{h_y}}\overrightarrow{h_t} + W_{\overleftarrow{h_y}}\overleftarrow{h_t} + b_y\right) \tag{19}$$

where $W_{\overrightarrow{h_y}}$ and $W_{\overleftarrow{h_y}}$ denote the weight matrices of forward and backward LSTM layers, respectively. b_y denotes the output layer bias. Moreover, $\sigma(\bullet)$ is a concatenating or an averaging function that combines both outputs.

Bi-LSTM structure allows the capture of important trends from data with forward and backward moving windows [50]. Thus, this study also investigates Bi-LSTM model for multivariate predictions for TWINS wind sensing board. LSTM and Bi-LSTM models require three-dimensional (3D) input as (batch-size, time steps, number of features). In this work, the input shape for these models is used as (512, 1, 4).

The DL architectures including MLP, LSTM, and Bi-LSTM have been implemented with three hidden layers of 16, 8, and 4 units. The parameters learned from the hidden layers of the DL models are lastly fed to the output layer, which comprises of two dense units with linear activation. The final dense units use linear activation function to yield the output variables. The number of inputs is four, from two wind-sensing boards (four input variables: B_LONG and B_TRANS of the remaining two boards). While, the number of outputs of the model is two, as the model is targeted for predicting variables for one board of TWINS boom.

5. Experimental configuration and evaluation

The overall methodology used for DL models is illustrated in figure 7.

The proposed DL models are trained with an unshuffled dataset. The raw data is fed to the models after pre-processing as explained in section 3. The activation functions are added in each successive layers of DL models except the last dense layer, which yields multivariate regression output. The Adam optimizer is employed for efficient training of the models. The optimizer allows the model to learn efficiently and yield accelerated and improved convergence towards minima. The learning rate was set to 0.0001 for effective learning of the models. The models are trained with a batch size of 512. The loss function used for the model is the mean squared error (MSE) given in equation (20). The training process is automatically controlled using early stopping callback, which stops the model when it starts overfitting on the data. The early stopping callback is used which monitors the validation error to avoid the overfitting problem. When the training process is stopped, the learned model parameters are saved. Lastly, the saved model is used to make predictions and then predicted data is inversely power transformed. The LSTM and Bi-LSTM models require 3D input shape (batch-size, time step, features), thus the input shape has been converted as per required shape. The models are fed with input features at one-time step per sample.

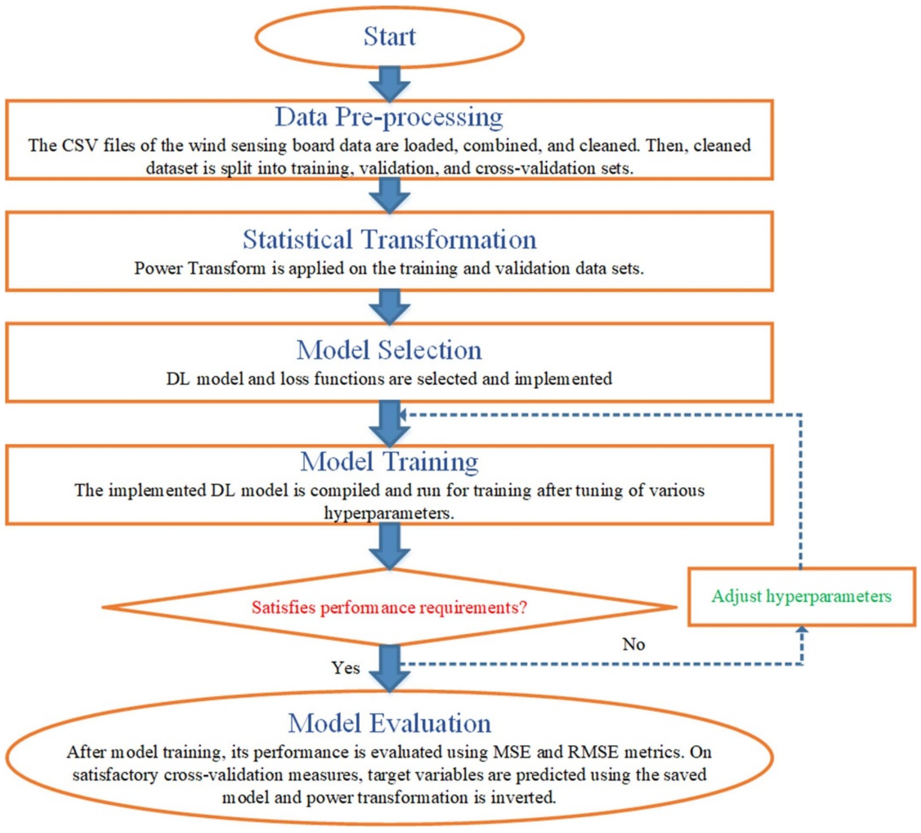

Start
Data Pre-processing
The CSV files of the wind sensing board data are loaded, combined, and cleaned. Then, cleaned dataset is split into training, validation, and cross-validation sets.
Statistical Transformation
Power Transform is applied on the training and validation data sets.
Model Selection
DL model and loss functions are selected and implemented
Model Training
The implemented DL model is compiled and run for training after tuning of various hyperparameters.
Satisfies performance requirements?
Adjust hyperparameters
No
Yes
Model Evaluation
After model training, its performance is evaluated using MSE and RMSE metrics. On satisfactory cross-validation measures, target variables are predicted using the saved model and power transformation is inverted.

Figure 7. Overview of methodology of DL models.

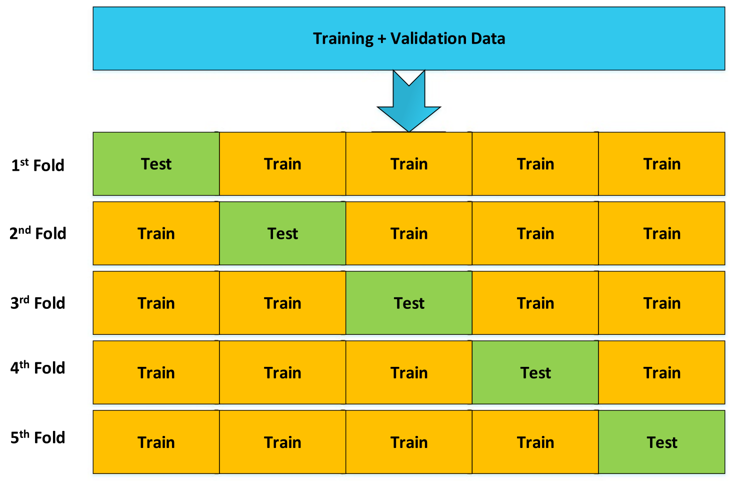


Figure 8. Grid search cross-validation used for ML models.

Regarding ML models (MPR, KNN, RF, and XGBoost) hyperparameters are obtained through the grid-search with the *k*-fold CV approach [51]. The grid-search CV approach allows to obtain most effective parameters based on *k*-folds. The grid-search CV approach is used to evaluate performance of ML models on unseen data or to find out how ML models would perform when these models are deployed for generating predictions in actual scenarios. We use five folds of data with the grid-search CV technique to obtain optimal parameters of the ML models. Figure 8 illustrates the general *k*-fold CV procedure used with the grid search approach. Moreover, the hypermeters obtained through the grid-search method are reported in table 1. The training and validation sets (90%) of the data are used for finding optimal parameters of the ML models including KNN, RF, and XGboost. After obtaining the optimal parameters, the models are trained using the

Table 1. Parameters of ML models obtained through grid-search CV.

Model	Parameters
MPR	Polynomial degree = 3
KNN	Neighbors = 150, leaf size = 20, $p = 2$
RF	Estimators = 25, maximum depth = 20, random state = 16
XGBoost	Estimators = 30, maximum depth = 15, learning rate = 0.1, objective = 'reg:squarederror'

training set (70%). Furthermore, these models are tested using the testing/CV (10%) so that their performance can be compared with the DL models.

The models are trained on a graphics processing unit (GPU) sever with 256 GB RAM and four GPU units. The DL models are implemented with Keras and Tensorflow environments and ML models are implemented with Scikit-learn library. Some essential python libraries such as Numpy, Pandas, Seaborn, and Matplotlib are employed for data preprocessing and plotting the achieved results. Moreover, the performance of the model is evaluated using RMSE and mean relative error (MRE) as expressed as follows:

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y_i})^2 \tag{20}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y_i})^2} \tag{21}$$

$$\mathrm{MRE} = \frac{1}{n}\sum_{i=1}^{n}\frac{y_i - \widehat{y_i}}{y_i}. \tag{22}$$

where y_i denotes the actual values, $\widehat{y_i}$ denotes the predicted values, and n denotes the number of samples.

6. Results and discussions

This section reports and discusses the results achieved with the proposed method. In this study, seven ML and DL methods were explored for multivariate predictions. For the applied methods, RMSE errors in BLONG-BTRANS variables and horizontal wind speed and angle are studied. The comparative analysis of the experimented methods allowed us to select an appropriate method which yields effective and consistent performance for all six wind sensing boards. The models have been tested for each one of the six wind sensing boards of two booms (at each time, only one board in one boom is predicted, using only the data from the other two boards in the same boom). The proposed models were efficiently trained for multivariate predictions of the six cases owing to the computing power of the GPU server, statistical transformation, and effective hyperparameters. Finally, the KNN, MLP, and LSTM models are able to generalize well on the dataset and able to yield consistent performance in all six cases as can be observed from table 2. The KNN model was found to be most suitable for the proposed soft wind sensing board owing to its high generalization performance, simplicity, and its lower computational cost compared to DL models (MLP and LSTM) which are computationally intensive due to the large number of parameters.

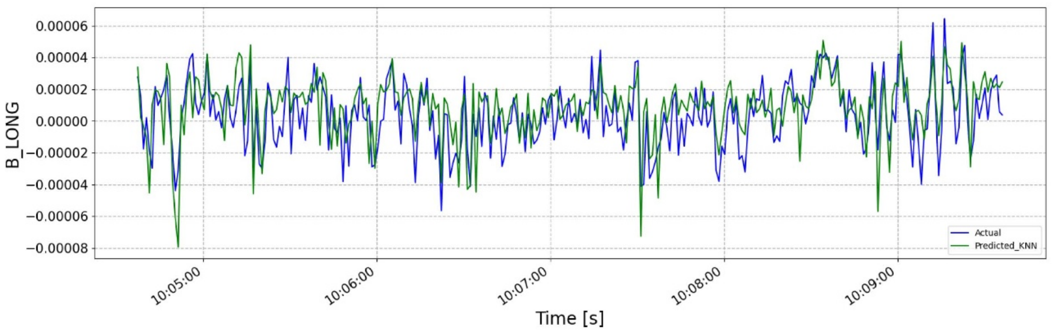
The achieved results for the board-1 of the BMY boom are depicted in figure 9, which depicts line plots of the board-1 for the data of 5 min. The plot depicts the intermediate variables B_LONG and B_TRANS, and shows how predicted values are close to the actual values for all three boards.

Figure 10(a) shows the comparison between velocity obtained from the ground truth data of the three boards and the IA, the velocity obtained from two boards' data and the RIA, as well as the velocity obtained from two boards and the one predicted board using a KNN model, fed into the IA.

As it can be observed from the figures, the wind velocity with RIA and two boards is quite different from the actual wind velocity. Moreover, the velocity obtained with two boards is highly fluctuating and in most instances, its magnitude is much greater than that of velocity obtained with three boards. Contrarily, the velocity obtained with the predicted data is quite similar to the original velocity as depicted in the figure. Similarly, recovered wind direction values with one board predicted are very close to the actual values compared to the values obtained from two boards and RIA which has a lot of fluctuations as shown in figure 10(b). A clear impact of the absence of BMY board-1 can be observed in the velocity and wind direction from two board's data which depicts higher fluctuations and has larger magnitude at some points

Table 2. RMSE between predicted and actual data of the wind sensor.

Model	BMY_1	BMY_2	BMY_3	BPY_1	BPY_2	BPY_3
PR	2.55569×10^{-5}	2.59896×10^{-5}	8.74659×10^{-5}	2.68655×10^{-5}	2.66953×10^{-5}	2.00411×10^{-5}
KNN	1.89449×10^{-5}	1.75557×10^{-5}	1.8101×10^{-5}	2.13752×10^{-5}	2.58112×10^{-5}	1.73513×10^{-5}
RF	1.93928×10^{-5}	1.79587×10^{-5}	1.84208×10^{-5}	2.16890×10^{-5}	4.72743×10^{-5}	1.76182×10^{-5}
XGBoost	1.98910×10^{-5}	1.83221×10^{-5}	1.86993×10^{-5}	2.30502×10^{-5}	4.04046×10^{-5}	1.77320×10^{-5}
MLP	1.96089×10^{-5}	1.82384×10^{-5}	1.85257×10^{-5}	2.33002×10^{-5}	2.55810×10^{-5}	2.55811×10^{-5}
LSTM	1.95829×10^{-5}	1.77569×10^{-5}	1.80511×10^{-5}	2.11124×10^{-5}	2.81874×10^{-5}	1.80584×10^{-5}
Bi-LSTM	1.94329×10^{-5}	1.92686×10^{-5}	1.78694×10^{-5}	2.04109×10^{-5}	3.73448×10^{-5}	1.75080×10^{-5}



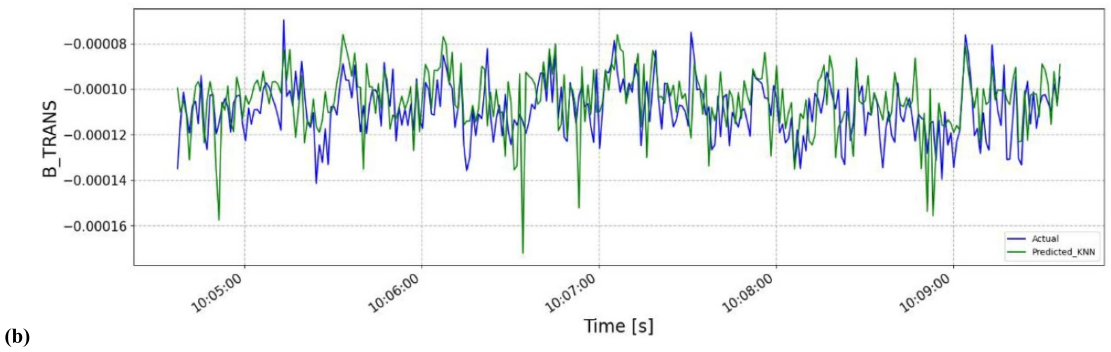


Figure 9. Actual versus predicted values for the wind sensing boards of BMY board-1 for 5 min (a) BLONG values and (b) BTRANS values. The blue and green lines seem to be close enough and following the trend in the data.

than the velocity and wind direction obtained from three boards. In contrast, the velocity with one board predicted, is quite similar to the ground truth velocity. Thus, the obtained results from the IA and one board predicted confirm the usefulness of the proposed method.

Figures 11(a) and (b) illustrate RMSE of the wind variables including wind velocity and wind angle for the total test data. The errors are computed between the wind values estimated from predicted using different models and the original test dataset from all three boards, using the IA in both cases. Additionally, these figures also depict the RMSE between the estimated values based on three boards with IA, and two boards with the RIA. The errors were computed by taking into account only the data from booms (BMY/BPY) that were selected as test data for obtaining the wind estimations from the intermediate variables, i.e. the selected boom is well positioned for the angle of the wind. In all cases, one of three boards is unavailable.

The comparative error analysis from figure 11, clearly shows that the applied models are helpful in recovering wind values and are better than the values obtained from two boards and RIA. Among the applied models, some models including KNN, MLP, and LSTM seem to perform consistently. However, PR depicts high error for board-1 of BMY boom while RF, XGBoost, and Bi-LSTM show high error in case of the board-3 of BPY boom. Similarly, the KNN, MLP, and LSTM models perform well for all six boards as can be observed from figure 11(b). While, PR, RF, XGBoost, and Bi-LSTM depict high wind direction error for at least one board out of the six boards.

Similarly, figures 12(a) and (b) depicts MRE of the wind variables including wind velocity and wind angle for the test set. It also shows that the proposed methods are effective in recovering the wind values from the relevant variables in the case of one missing board. Furthermore, it confirms the superiority of KNN, MLP, and LSTM models over the other employed methods.

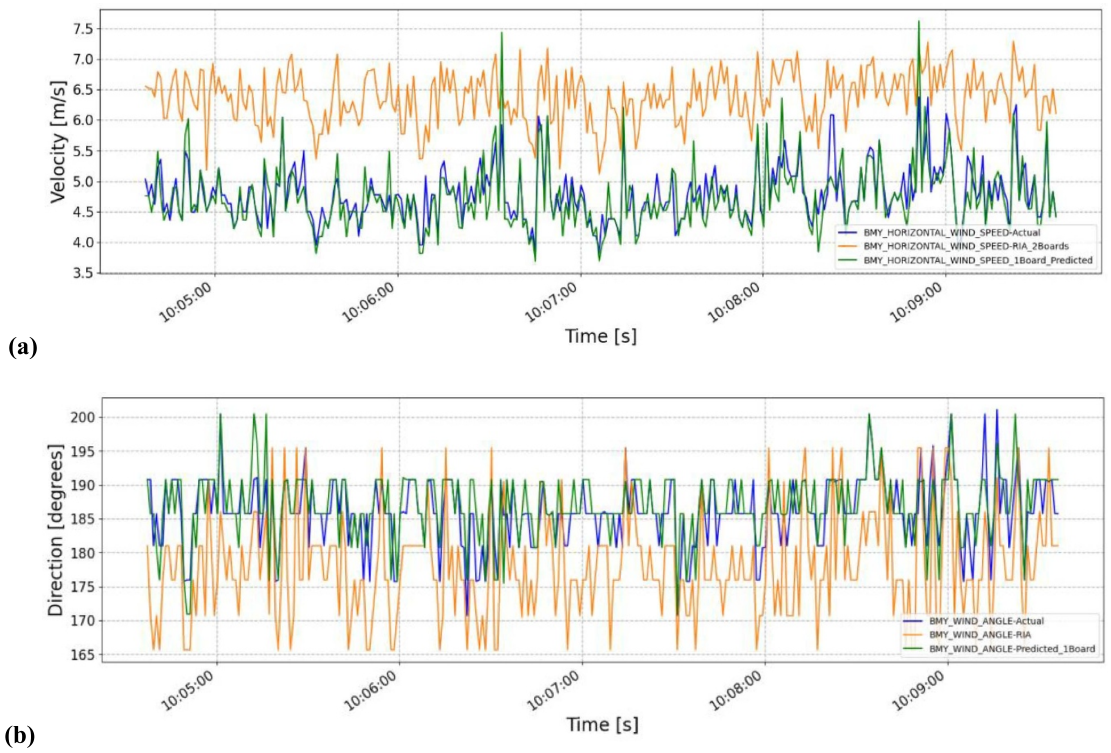


Figure 10. Comparison between the values obtained using the inverse algorithm with the original three boards' data, the restricted inverse algorithm with the original data of only two boards, and velocity obtained using the data predicted for one missing board and the remaining boards with the complete IA of (a) horizontal velocity and (b) wind direction.

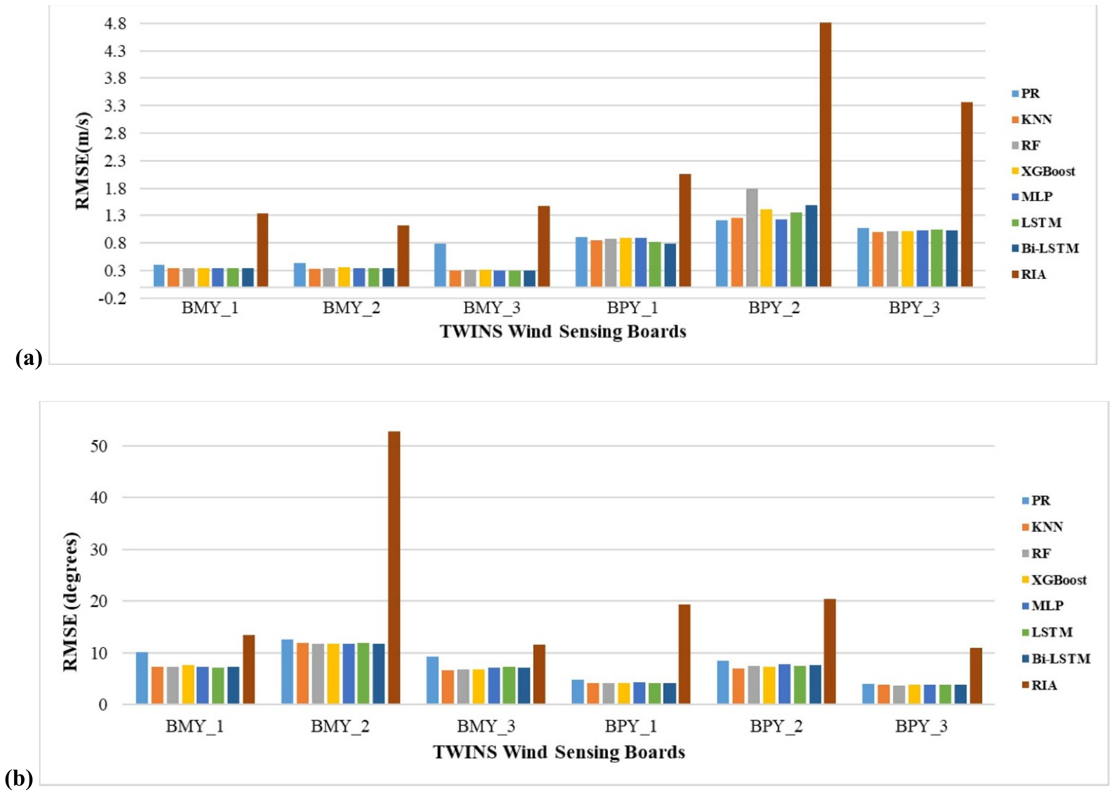


Figure 11. RMSE comparison between the applied models and RIA of (a) wind velocity and (b) wind direction.

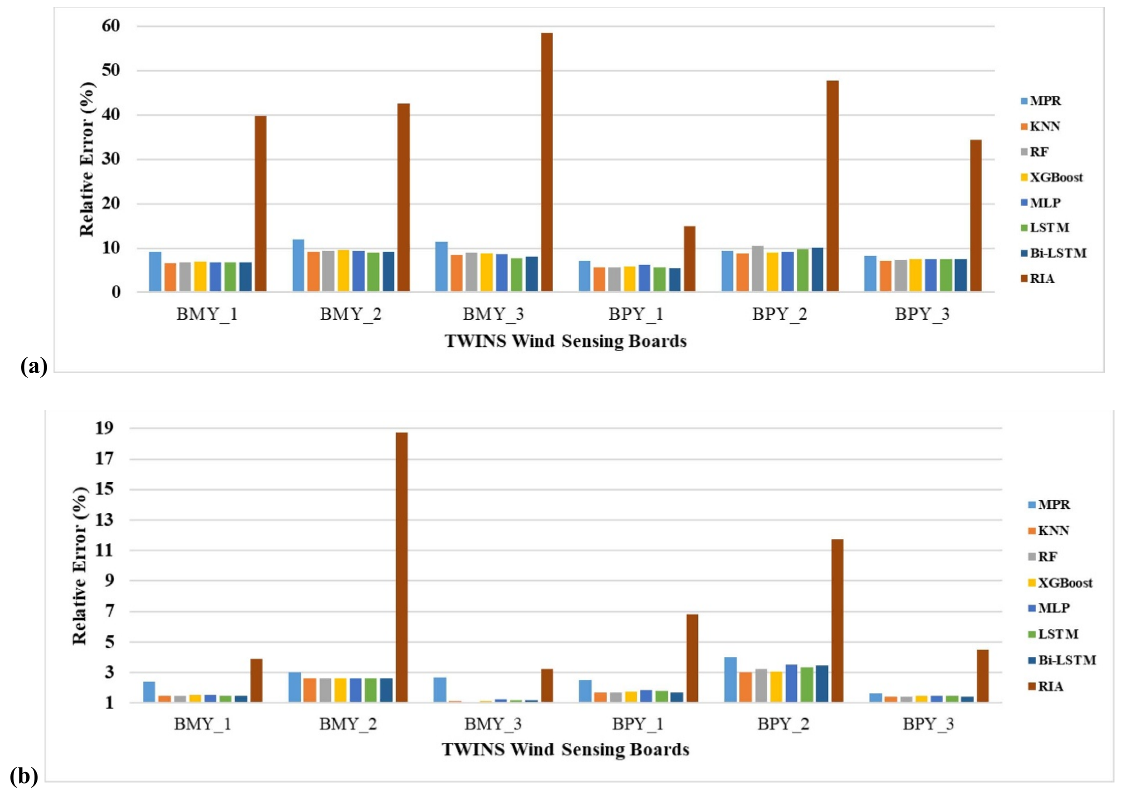


Figure 12. MRE comparison between the applied models and RIA of (a) wind velocity and (b) wind direction.

The performance comparison allows the KNN model to stand out among all the models based on its effective performance and simple architecture. In fact, MLP and LSTM also demonstrate an effective performance and have yielded similar as that of KNN. Among these models, KNN would be an obvious choice owing to its less-complex architecture and lower computational cost.

Considering the nature of data, the statistical transformation was essential to avoid bias and complexity in the training process. Some methods such as PR and KNN do not essentially require data transformation and these methods are able to work without data transformation. Nonetheless, by applying power transformation, the data were converted to a Gaussian-like distribution and these transformed data allowed for a better and smoother training process. For a large and complex time-series dataset, it becomes crucial to apply statistical transformations to achieve targeted results using ML and DL models. Moreover, the use of grid-search CV approach resulted in a smooth training process without the overfitting problem. Thus, various models with the necessary statistical data transformations were efficiently trained after applying the appropriate data transformation. Comparatively, the models were able to perform better on the data of BMY boom as it can be observed from the error reduced in table 2.

The comparative analysis of the seven data-driven models revealed that the simple ML model like KNN was able to yield similar performance as that of LSTM and MLP models. We expected the LSTM models to work far better than the other methods, but this did not occur. On the other hand, different variables were input to the algorithms, some more or less correlated with the variables of the two boards of the same boom that are used to predict the missing values. This is why in some cases KNN and MLP might work better than Bi-LSTM and LSTM, in some cases the other way around. In LSTM based methods, time memory could play an important role, but given the results it does not. Additionally, it must be noted that the correlation between all the variables clearly depends on the wind magnitude and direction.

Considering the multivariate missing data problem, KNN has proven its effectiveness in multivariate numerical data imputation compared to powerful DL algorithms [52]. Depending upon the nature of numerical data, KNN has been able to yield better or close enough performance to the complex models in various studies [53–56]. Therefore, it is often preferred over complex models for numerical data imputation owing to its competitive performance, deterministic nature, and low computational requirements.

It must be noted that, even though all ML and DL models give comparable prediction metrics, in case the calculation should be made in situ (on Mars surface, using the very limited computation resources of the rover or lander) the trained DL model might be more useful, since they require less computational resources on edge to obtain the inferences.

The complex aerodynamics generated by the wind on the booms requires an inversion algorithm based on extensive calibration in the wind tunnel. This means that there are no analytical models or approximations describing the sensor outputs as a function of the incoming wind. On the other hand, having three boards on each boom typically guarantees that at least two of them sense wind very well. This means that from the aerodynamical point of view, a certain correlation between the output signals is to be expected. This is one of the key factors explaining the fact that the algorithms presented in this work are capable of producing good predictions.

7. Conclusions

In this paper, a comparative analysis of seven ML and DL models was carried out to predict intermediate variables from a complex sensor, such as the wind sensor from the TWINS instrument. The study is aimed at improving the resilience of sensors using the proposed methods. The models were trained with the statistically transformed data of two wind-sensing boards to predict the data for the third board. The evaluation based on RMSE of the outputs demonstrated significant potential of the KNN, MLP, and LSTM models in terms of predictions. Nonetheless, the KNN model is the preferable choice for the soft wind sensing board owing to its simple architecture, lower computational cost, and robust performance for all six cases. The performance of the proposed model to obtain wind values was evaluated based on the results of the IA. The comparative error analysis demonstrated good fit between wind values obtained through original and predicted data. The wind recovery errors for the horizontal velocity are reduced by a factor between 2.43 and 4.78, compared with the situation of using only two boards, and restricting the IA to them. Similarly, the error for the wind direction is reduced by a factor between 1.74 and 4.71.

This approach can be applied to other sensors working with complex intermediate variables sets and calibration data.

Data availability statement

The data that has been used is confidential.

Acknowledgments

Funded in part by MICINN, Grant Nos. PID2021-126719OB-C41 and PID2021-126719OB-C42. D K acknowledges FI-SDUR Grant No. 2021 FI SDUR 00328 from the Generalitat de Catalunya.

Funding

This work has been partially supported by the Spanish Ministry MICINN under Grant Nos. PID2021-126719OB-C41 and PID2021-126719OB-C42.

Authorship contribution statement

Conceptualization, M D-P, D K, E S-C; Software programming, D K, L M, S N; TWINS Data curation, J T, M M, J G-E, L M, S N, J R-M; Investigation, all authors; Validation M D-P, E S-C, S N, J G-E, J R-M; Supervision, M D-P, E S-C, J -R-M; Writing first draft, D K, M D-P, E S-C; Review & editing: all authors.

Conflict of interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

ORCID iDs

Dileep Kumar https://orcid.org/0000-0002-6211-1078
Manuel Dominguez-Pumar https://orcid.org/0000-0001-5439-7953
Elisa Sayrol-Clols https://orcid.org/0000-0002-0526-9733
Josefina Torres https://orcid.org/0000-0003-1035-6740
Mercedes Marín https://orcid.org/0000-0003-2328-1303

Javier Gómez-Elvira https://orcid.org/0000-0002-9068-9846
Luis Mora https://orcid.org/0000-0002-8209-1190
Sara Navarro https://orcid.org/0000-0001-8606-7799
Jose Rodríguez-Manfredi https://orcid.org/0000-0003-0461-9815

References

[1] Banfield D *et al* 2019 InSight auxiliary payload sensor suite (APSS) *Space Sci. Rev.* **215** 1–33
[2] Gómez-Elvira J *et al* 2012 REMS: the environmental sensor suite for the Mars Science Laboratory rover *Space Sci. Rev.* **170** 583–640
[3] Rodriguez-Manfredi J A *et al* 2021 The Mars environmental dynamics analyzer, MEDA. A suite of environmental sensors for the Mars 2020 mission *Space Sci. Rev.* **217** 1–86
[4] Hueso R *et al* 2023 Convective vortices and dust devils detected and characterized by Mars 2020 *J. Geophys. Res. Planets* **128** e2022JE007516
[5] Gõmez-Elvira J *et al* 2014 Curiosity's Rover Environmental Monitoring Station: overview of the first 100 sols *J. Geophys. Res. Planets* **119** 1680–8
[6] Bridges N T, Gómez-Elvira J, Marin M, Navarro S, Torres J, Richardson M I, Battalio J M, Guzewich S D and Sullivan R 2017 Winds measured by the Rover Environmental Monitoring Station (REMS) during the Mars Science Laboratory (MSL) rover's Bagnold Dunes Campaign and comparison with numerical modeling using MarsWRF *Icarus* **291** 203–31
[7] Oehmcke S, Zielinski O and Kramer O 2018 Input quality aware convolutional LSTM networks for virtual marine sensors *Neurocomputing* **275** 2603–15
[8] Saroj A J, Guin A and Hunter M 2021 Deep LSTM recurrent neural networks for arterial traffic volume data imputation *J. Big Data Anal. Transp.* **3** 95–108
[9] Verma H and Kumar S 2019 An accurate missing data prediction method using LSTM based deep learning for health care *ICDCN '19: Proc. 20th Int. Conf. on Distributed Computing and Networking* (*4 January 2019*) (*ACM Int. Conf. Proc. Series*) pp 371–6
[10] Banfield D, Spiga A, Newman C, Forget F, Lemmon M, Lorenz R, Murdoch N, Viudez-Moreiras D, Pla-Garcia J and Garcia R F 2020 The atmosphere of Mars as observed by InSight *Nat. Geosci.* **13** 190–8
[11] Pan H, Su T, Huang X and Wang Z 2021 LSTM-based soft sensor design for oxygen content of flue gas in coal-fired power plant *Trans. Inst. Meas. Control* **43** 78–87
[12] Ren L, Wang T, Laili Y and Zhang L 2022 A data-driven self-supervised LSTM-DeepFM model for industrial soft sensor *IEEE Trans. Ind. Inform.* **18** 5859–69
[13] Zaidan M A *et al* 2020 Intelligent calibration and virtual sensing for integrated low-cost air quality sensors *IEEE Sens. J.* **20** 13638–52
[14] Khaldi B, Harrou F, Benslimane S M and Sun Y 2021 A data-driven soft sensor for swarm motion speed prediction using ensemble learning methods *IEEE Sens. J.* **21** 19025–37
[15] Vallejo M, De La Espriella C, Gómez-Santamaría J, Ramírez-Barrera A F and Delgado-Trejos E 2019 Soft metrology based on machine learning: a review *Meas. Sci. Technol.* **31** 32001
[16] Schmidhuber J 2015 Deep learning in neural networks: an overview *Neural Netw.* **61** 85–117
[17] LeCun Y, Bengio Y and Hinton G 2015 Deep learning *Nature* **521** 436–44
[18] Najafabadi M M, Villanustre F, Khoshgoftaar T M, Seliya N, Wald R and Muharemagic E 2015 Deep learning applications and challenges in big data analytics *J. Big Data* **2** 1–21
[19] Meritxell G O, Sierra B and Ferreiro S 2022 On the evaluation, management and improvement of data quality in streaming time series *IEEE Access* **10** 81458–75
[20] Zhou L, Pan S, Wang J and Vasilakos A V 2017 Machine learning on big data: opportunities and challenges *Neurocomputing* **237** 350–61
[21] Ahmed Z 2022 Hysteresis compensation in temperature response of fiber Bragg grating thermometers using dynamic regression *Sens. Actuators* A **347** 113872
[22] Zhao J and Ye F 2019 Where ThermoMesh meets ThermoNet: a machine learning based sensor for heat source localization and peak temperature estimation *Sens. Actuators* A **292** 30–38
[23] Kim J, Shin W, Hong S, Jeong Y, Jung G, Choi W Y, Kim J J, Park B G and Lee J H 2023 A novel pathway to construct gas concentration prediction model in real-world applications: data augmentation; fast prediction; and interpolation and extrapolation *Sens. Actuators* B **382** 133533
[24] Tonezzer M, Kim J H, Lee J H, Iannotta S and Kim S S 2019 Predictive gas sensor based on thermal fingerprints from Pt-SnO_2 nanowires *Sens. Actuators* B **281** 670–8
[25] Li D, Wang Y, Wang J, Wang C and Duan Y 2020 Recent advances in sensor fault diagnosis: a review *Sens. Actuators* A **309** 111990
[26] Dixneuf P, Errico F and Glaus M 2021 A computational study on imputation methods for missing environmental data (arXiv:2108.09500)
[27] Sangari S and Ray H E 2021 Evaluation of imputation techniques with varying percentage of missing data (arXiv:2109.04227)
[28] Zheng S and Charoenphakdee N 2022 Diffusion models for missing value imputation in tabular data (arXiv:2210.17128)
[29] Nebot O B 2021 *Artificial Intelligence Applied to Improve Resilience in the Inverse Algorithms of Mars Wind Sensors* (cnica de Universitat Politècnica de Catalunya)
[30] Sheng Y 2021 *Design of Algorithms for Improving Resilience of Sensors in Space Exploration* (cnica de Universitat Politècnica de Catalunya)
[31] Domínguez M, Jiménez V, Ricart J, Kowalski L, Torres J, Navarro S, Romeral J and Castañer L 2008 A hot film anemometer for the Martian atmosphere *Planet. Space Sci.* **56** 1169–79
[32] Atienza M T, Kowalski L, Gorreta S, Jiménez V and Domínguez-Pumar M 2018 Thermal dynamics modeling of a 3D wind sensor based on hot thin film anemometry *Sens. Actuators* A **272** 178–86
[33] Yeo I N K and Johnson R A 2000 A new family of power transformations to improve normality or symmetry *Biometrika* **87** 954–9
[34] Bisandu D B, Moulitsas I and Filippone S 2022 Social ski driver conditional autoregressive-based deep learning classifier for flight delay prediction *Neural Comput. Appl.* **34** 8777–802
[35] Dinh P T and Park M 2021 R-EDoS: robust economic denial of sustainability detection in an SDN-based cloud through stochastic recurrent neural network *IEEE Access* **9** 35057–74

[36] Wei J, Chen T, Liu G and Yang J 2016 Higher-order multivariable polynomial regression to estimate human affective states *Sci. Rep.* **6** 1–13
[37] Sahli H 2020 *An Introduction to Machine Learning* (Springer) (https://doi.org/10.1002/9781119720492.ch7)
[38] Emmanuel T, Maupong T, Mpoeleng D, Semong T, Mphago B and Tabona O 2021 A survey on missing data in machine learning *J. Big Data* **8** 1–37
[39] Breiman L 2001 Random forests *Mach. Learn.* **45** 5–32
[40] Hernández-del-olmo F, Gaudioso E, Duro N and Dormido R 2019 Machine learning weather soft-sensor for advanced control of wastewater treatment plants *Sensors* **19** 3139
[41] Nkulikiyinka P, Yan Y, Güleç F, Manovic V and Clough P T 2020 Prediction of sorption enhanced steam methane reforming products from machine learning based soft-sensor models *Energy* AI **2** 100037
[42] Azen S P and Naeve P 2006 Computational statistics and data analysis (CSDA) *Encycl. Stat. Sci.* **38** 367–78
[43] Chen T, He T, Benesty M, Khotilovich V, Tang Y, Cho H and Chen K 2015 Xgboost: extreme gradient boosting. R package version 0.4–2 *OS Indep.* pp 1–4
[44] Yang S, Wu J, Du Y, He Y and Chen X 2017 Ensemble learning for short-term traffic prediction based on gradient boosting machine *J. Sens.* **2017** 1–15
[45] Kazemi P, Steyer J P, Bengoa C, Font J and Giralt J 2020 Robust data-driven soft sensors for online monitoring of volatile fatty acids in anaerobic digestion processes *Processes* **8** 67
[46] Eskandarian S, Bahrami P and Kazemi P 2017 A comprehensive data mining approach to estimate the rate of penetration: application of neural network, rule based models and feature ranking *J. Pet. Sci. Eng.* **156** 605–15
[47] Hecht-Nielsen R 1992 Theory of the backpropagation neural network *Neural Networks for Perception: Computation, Learning, and Architectures* (Elsevier) pp 65–93
[48] Hochreiter S and Schmidhuber J 1997 Long short-term memory *Neural Comput.* **9** 1735–80
[49] Zhu X, Hao K, Xie R and Huang B 2021 Soft sensor based on eXtreme gradient boosting and bidirectional converted gates long short-term memory self-attention network *Neurocomputing* **434** 126–36
[50] Yang Z, Jia R, Wang P, Yao L and Shen B 2022 Supervised attention-based bidirectional long short-term memory network for nonlinear dynamic soft sensor application *ACS Omega* **8** 4196–208
[51] Pedregosa F *et al* 2011 Scikit-learn: machine learning in python *J. Mach. Learn. Res.* **12** 2825–30
[52] Lalande F and Doya K 2022 Numerical data imputation: choose kNN over deep learning *Int. Conf. on Similarity Search and Applications* (Springer) pp 3–10
[53] Lachaab M and Omri A 2023 Machine and deep learning-based stock price prediction during the COVID-19 pandemic: the case of CAC 40 index *EuroMed J. Bus.* (https://doi.org/10.1108/EMJB-05-2022-0104)
[54] Dwivedi D N and Patil G 2023 Climate change: prediction of solar radiation using advanced machine learning techniques *Visualization Techniques for Climate Change with Machine Learning and Artificial Intelligence* (Elsevier) pp 335–58
[55] Choudhury A, Middya A I and Roy S 2022 A comparative study of machine learning and deep learning techniques in forecasting air pollution levels *Proc. Int. Conf. on Data Science and Applications: ICDSA 2021* vol 1 (Springer) pp 607–19
[56] Khazaee Poul A, Shourian M and Ebrahimi H 2019 A comparative study of MLR, KNN, ANN and ANFIS models with wavelet transform in monthly stream flow prediction *Water Resour. Manage.* **33** 2907–23