1st Latin-American Workshop on Structural Health Monitoring
LATAM-SHM 2023
December, 5-8 2023

DETECTION OF LOSSY BOLTS IN A JACKET-TYPE WIND TURBINE SUPPORT USING A VIBRATION-ONLY RESPONSE STRATEGY BASED ON ACCELEROMETER DATA

RICARDO PRIETO-GALARZA[*,‡], CHRISTIAN TUTIVÉN[*,†] AND YOLANDA VIDAL[*,◇]

[*] Control, Data, and Artificial Intelligence, CoDAlab
Department of Mathematics, Escola d'Enginyeria de Barcelona Est, EEBE
Universitat Politècnica de Catalunya, UPC
Campus Diagonal-Besós (CDB) 08019, Barcelona, Spain
e-mail: {ricardo.prieto,yolanda.vidal}@upc.edu - Web page: https://www.upc.edu/es

[†]Mechatronics Engineering
Faculty of Mechanical Engineering and Production Science, FIMCP
Escuela Superior Politécnica del Litoral, ESPOL
Campus Gustavo Galindo Km. 30.5 Vía Perimetral, P.O. Box 09-01-5863, Guayaquil, Ecuador
Phone number: +593 9 9103 5259,
e-mail: cjtutive@espol.edu.ec - Web page: https://www.espol.edu.ec

[‡]Universidad Ecotec
Km. 13.5 Samborondón, Samborondón, EC092302 Ecuador.

[◇]Institute of Mathematics (IMTech)
Universitat Politècnica de Catalunya (UPC)
Pau Gargallo 14, 08028 Barcelona, Spain
e-mail: yolanda.vidal@upc.edu - Web page: https://imtech.upc.edu/en

Key words: wind turbine, structural health monitoring, damage detection, jacket structure

Abstract. The early detection of damage in wind turbine structures is of crucial importance to ensure the safety and efficiency of wind farms. This study specifically addresses the problem of detecting a common damage: loose bolts, which can have a significant impact on the performance and lifespan of a jacket-type structure. This work focuses on the structural health monitoring of jacket-type support used by offshore wind turbines. A vibration-only response strategy based on accelerometer data is specifically proposed to detect loose bolts in the wind turbine jacket using an anomaly detection model. The methodology consists of two training phases using only healthy data: training a generative adversarial network (GAN) and training an encoder based on the learned GAN model. Through the GAN network training process, a generator, and a critic are obtained. The encoder is then trained to map healthy samples to a latent vector, placing the data at points in the latent space that correspond to the healthy state of the input data. Once the encoder training is complete, the encoder maps the input sample space to the latent space, and the generator maps the latent space back to the initial space. In the case of a healthy input, this mapping process should closely resemble the original input sample. However, when damaged-state input samples are used, the model output does

not resemble the input. To identify anomalies, the reconstruction error and a comparison of the residual error of the critic properties are employed as the final two loss functions. The proposed strategy has been validated through laboratory experiments on a down scaled model. The deep learning models have proven to be an effective technique for the early detection of loose bolts in jacket-type structures of wind turbines. This approach can significantly contribute to improving the safety and performance of wind farms by enabling timely and efficient intervention in the event of possible structural failures.

1 INTRODUCTION

In recent years, the global utilization of wind energy has seen a substantial rise [1], with an increasing installation of offshore wind turbines (WTs) to harness more consistent and stronger winds. However, to sustain this growth, it is imperative to concurrently reduce operational and maintenance costs. In this context, structural health monitoring (SHM) solutions have emerged as a means to provide early alerts, thereby reducing maintenance expenses and extending the lifespan of turbines.

Various types of supports are used for WTs, and this study specifically focuses on jacket-type structures, which are preferred for locations with significant water depths and unfavorable soil conditions [2]. To enhance SHM in WTs, the integration of artificial intelligence (AI) is a promising avenue as it aids in predicting component damage. Unsupervised learning methods have garnered significant attention recently, enabling the analysis of extensive sensor data to identify patterns indicative of damage. For instance, in a recent work, Feijóo et al. utilized an unsupervised learning approach based on autoencoders (AEs) to scrutinize vibration sensor data from WTs [3]. The AE model is employed to extract pertinent features from sensor data, facilitating the detection of structural anomalies and thus providing early warnings for potential issues. Another notable approach, as suggested by Mao et al. [4], involves the use of generative adversarial networks (GANs) for SHM in offshore WTs, significantly enhancing the capacity to identify damage and anomalies. Their study utilized GANs to analyze vibration data from turbines, generating images representing turbines in various states. These images were subsequently employed to train a supervised classification model for the detection of damage.

This study employs a combined GAN and AE model approach for early damage detection in WT jacket supports through unsupervised learning. Inspired by Schlegl et al.'s work on medical image anomaly detection [5], the research aims to develop an unsupervised deep learning method for anomaly detection in experimental jacket structures using vibration data exclusively. The methodology involves simulating wind-induced rotor vibration via Gaussian white noise, collecting accelerometer data, preprocessing it into the time-frequency domain, normalizing it, training GAN and AE networks on healthy data, and testing the damage detection approach using loose bolts at four locations. Key contributions include i) relying solely on healthy data, ii) utilizing time-frequency feature extraction based on the Wigner-Ville distribution (WVD), iii) and operating with only vibration-response data, assuming the unknown input excitation signal is wind, waves, or currents.

The subsequent sections of this article are organized as follows. Section 2 elaborates on the development of the proposed methodology, Section 3 presents and discusses the obtained

results, and, finally, Section 4 summarizes the conclusions derived from the study.

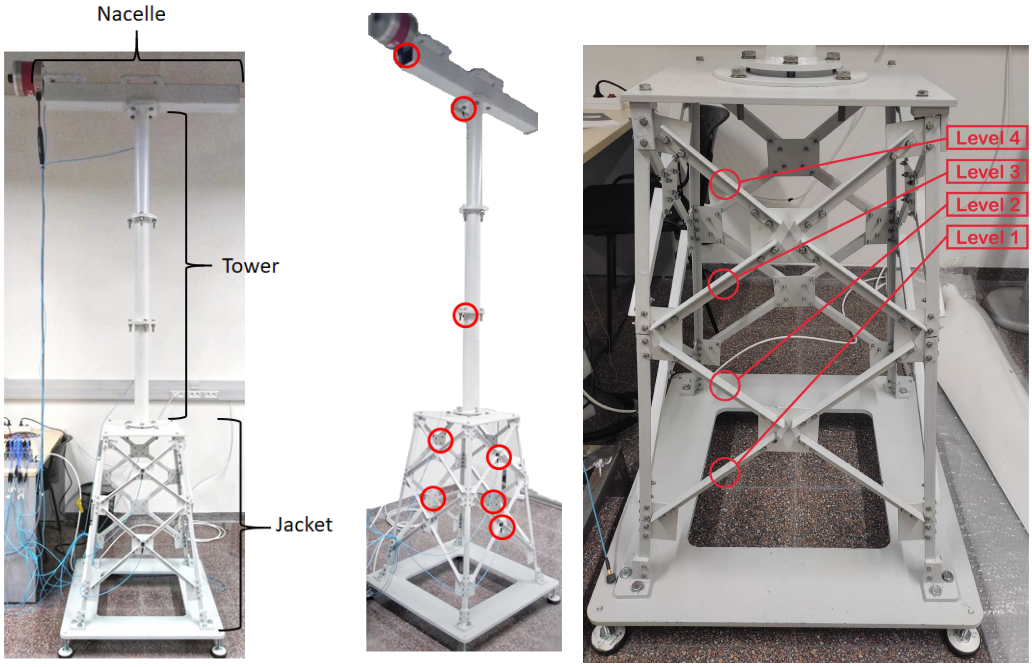
2 METHODOLOGY

The methodology collects experimental data across 5 structural conditions - healthy and loose bolt stats at four jacket support levels. The data is partitioned into training, validation and testing sets before transforming into the time-frequency domain using the WVD, converting into tensors. The tensors are normalized before training on the WGAN architecture and AE using only the healthy training data.

2.1 Experimental Setup

The utilized WT structure in this study is a scaled-down version, as depicted in Figure 1 (left).

Figure 1: Representation of the scaled-down structure (left). Placement of the sensors (mid). Positioning of the four levels within the jacket substructure (right).



The structure is composed of a jacket support, a tower, and a nacelle (simulated using a beam and an inertial shaker to model wind-induced rotor vibrations), measuring a height of 2.7 meters. At strategic locations on the turbine, eight triaxial acceleration sensors are positioned to collect three signals (namely, accelerations along the x-axis, y-axis, and z-axis), resulting in a total of 24 acceleration measurements, as shown in Figure 1 (mid). Furthermore, the turbine base is segmented into four levels, as depicted in Figure 1 (right).

WTs operate in three distinct wind speed regions. To simulate various wind speeds, a shaker is employed to induce vibrations in the structure, modifying the amplitude of the white noise signal (scaling by factors of 0.5, 1, and 2). This study focuses solely on the operational region with higher wind speeds, resulting in a white noise amplitude of 2. This choice is due to

the turbine's increased vulnerability to damage under higher wind conditions. For additional details on the experimental arrangement, refer to [6].

Loose bolts at 4 different levels are the damage cases studied in this research. The levels are according to those segmented in the Figure 1 (right).

2.2 Data Collection

A total of 125 experimental rounds are conducted, with 45 involving structures in a healthy condition and 80 involving structures with various types of damage. The distribution of experiments conducted for various structural conditions is detailed in Table 1.

Table 1: Matrices obtained for each structural state.

Level	Structural State	Matrix name	No. of experiments	Matrix dimensionality
	Healthy	X^0	45	4,459,365 × 24
1	Loose Bolt - Level 1	X^1	20	1,981,940 × 24
2	Loose Bolt - Level 2	X^2	20	1,981,940 × 24
3	Loose Bolt - Level 3	X^3	20	1,981,940 × 24
4	Loose Bolt - Level 4	X^4	20	1,981,940 × 24

Each experiment has a duration of 60 seconds, with a sampling rate of approximately 1651.616 Hz. Consequently, within each experiment, a total of 99097 data points are recorded for each of the 24 sensors. Thus, the structure of the data matrix can be expressed as follows:

$$\begin{bmatrix} x_{11} & x_{12} & \dots & x_{1N} \\ \vdots & \vdots & \ddots & \vdots \\ x_{21} & x_{22} & \dots & x_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ x_{M1} & x_{M2} & \dots & x_{MN} \end{bmatrix} \in \mathcal{X}_{M\times N}(\mathbb{R}), \tag{1}$$

where $M = 99097$ represents the total number of samples in each experiment, $N = 24$ represents the number of sensors, and $\mathcal{X}_{M\times N}$ denotes the vector space of a matrix in $\mathbb{R}$ with M rows and N columns.

Each structural state represents the aggregation of all experiments conducted under that specific condition and is denoted as X, differentiated by a superscript. Specifically, X^0 represents the matrix composed of 45 experiments with only healthy data, while X^1, X^2, X^3, and X^4 are matrices containing data related to structural damage (loose bolt) at each of the four damage levels, with 20 experiments conducted at each level. Table 1 provides an overview of the characteristics and dimensions of the matrices for each experiment and structural state.

2.3 Data Split: Train, Validation and Test Sets

In this work, the deep learning models, WGAN and AE, are exclusively trained using healthy data. Matrix X^0 is divided into three distinct subsets, namely X^0_{train}, X^0_{val}, and X^0_{test}, utilizing the following distribution: *i*) Training set: Comprising 80% of the data, equivalent to 36 experiments. *ii*) Validation set: Representing 11% of the data, equivalent to 5 experiments.

iii)Test set: Accounting for the remaining 9%, equivalent to 4 experiments. On the other hand, all experiments X^1, X^2, and X^3 are added to the test set.

The training set is utilized for the initial training of the model. The validation set plays a crucial role in the evaluation process by offering an unbiased means to validate any adjustments to hyperparameters. Finally, the test set is reserved for the ultimate evaluation of the model's accuracy. This evaluation is conducted once the model is fully trained and measures its performance on unseen data.

2.4 Feature Engineering

Feature engineering involves the creation of new features from the original ones during the preprocessing stage. In this work, this process is divided into two main parts. Initially, the data undergo reshaping, and subsequently, a transformation is applied using the WVD, which converts matrices into tensors.

In pursuit of rapid damage detection, the dataset is partitioned into 826 samples for each of the 24 sensors, corresponding to a detection time of 0.5 seconds. Matrix segments, previously in the time domain, are transformed into sub-tensors in the time-frequency domain using WVD. Each resulting sub-tensor measures $826 \times 826 \times 24$. However, due to the segmentation process, the last sub-tensor contains incomplete data and is thus omitted. As a result, a complete experiment consists of 119 sub-tensors, yielding a total size of $119 \times 826 \times 826 \times 24$ after feature engineering. This segmentation and feature engineering process is applied to matrices X^0_{train}, X^0_{val}, X^0_{test}, X^1, X^2, X^3, and X^4.

2.5 Normalization

Normalization is used to standardize the dataset on a common scale to enhance the training by ensuring that all features are of equal importance. The chosen normalization method is min-max within the range [-1, 1]. To perform it, the tensor X^0_{train} is selected as the source, from which both the maximum and minimum values are extracted, and employed to normalize all the other samples, including X^0_{val}, X^0_{test}, X^1, X^2, X^3, X^4.

To further enhance the methodological process, a reduction in tensor size is executed to accelerate the training speed. The interpolation technique applied is nearest neighbors. The procedure involves computing a reduction factor, which represents the ratio between the original size and the desired size. In this study, the reduction factor is determined to be 12.90. Therefore, the tensor is resampled every 13 coefficients along its various axes. Consequently, the final tensor shape for input into the deep learning models becomes $119 \times 64 \times 64 \times 24$.

2.6 WGAN Architecture

Schlegl et al. [5] propose a two-stage training process: (1) WGAN training and (2) encoder training based on the trained WGAN model. The primary objective of stage (1) is for the WGAN network to acquire the capability to generate tensors closely resembling healthy data. Conversely, stage (2) is aimed at constructing an effective latent space to reproducing healthy data. The combination of these two stages forms a composite model for anomaly detection.

The Wasserstein generative adversarial network is an unsupervised deep learning model with two neural networks - a generator and a critic. It operates in two stages to generate tensors resembling the training data. The Wasserstein distance [7], instead of cross-entropy loss, is

used to quantify the difference between the probability distributions of real $P(r)$ and generated $P(g)$ samples. Using the Wasserstein distance can improve training and generate higher quality samples compared to traditional GANs.

In the generation phase, the WGAN model aims to fool the critic into perceiving its generated tensors as authentic by minimizing $P(g)$ and $P(r)$. To improve generalization and prevent overfitting, early stopping is used for epoch selection, resulting in improved generated tensors and downstream segmentation [8]. Early stopping is also implemented in the AE subsection. The generator exclusively utilizes a 256-point random data tensor from the latent space, employing a four-layer architecture to address the complexity of the problem. A comprehensive description of these layers can be found in Table 2.

The initial layer, denoted as lg_1, serves as the input layer. Both prior to and following the second layer, lg_2, a 2x upsampling operation is applied. Layers lg_2 and lg_3 employ a leaky ReLU activation function and incorporate batch normalization with 128 and 64 units, respectively. Finally, the output layer, lg_4, features a batch normalization of 64 units and utilizes a hyperbolic tangent activation function, because in this way the generated tensors have the same range as the ones used as input (recall that the normalization used scales the data to [-1, 1]).

Table 2: Generator stage architecture in the WGAN.

Name	**Type**	**Output shape**	**Kernel**	**Stride**	**Padding**
lg_1	Linear	[1,32768]	n/a	n/a	n/a
lg_2	Conv2d	[128,32,32]	3	1	1
lg_3	Conv2d	[64,64,64]	3	1	1
lg_4	Conv2d	[24,64,64]	3	1	1

The generator's loss function is defined by $-\frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))$, where f_w represents the critic's loss function, g_θ denotes the generator's output, $z^{(i)}$ stands for the i-th latent vector, and m denotes the number of generated tensors.

On the contrary, the critic employs a scoring strategy based on the calculation of the Wasserstein distance. This calculation aims to maximize the separation between the probability distributions $P(g)$ and $P(r)$, effectively measuring their proximity. The proposed critic architecture for tensor generation during training encompasses 5 layers. A detailed description of these layers is provided in Table 3. In particular, layers lc_1, lc_2, lc_3, and lc_4 employ a leaky ReLU activation function, a dropout layer with a rate of 0.25, and batch normalization, with the output layers matching the number of units in each layer, respectively. Finally, the layer lc_5 represents the final layer, producing a single output value referred to as the score.

Table 3: Critic stage architecture in the WGAN.

Name	**Type**	**Output shape**	**Kernel**	**Stride**	**Padding**
lc_1	Conv2d	[16,32,32]	3	2	1
lc_2	Conv2d	[32,16,16]	3	2	1
lc_3	Conv2d	[64,8,8]	3	2	1
lc_4	Conv2d	[128,4,4]	3	2	1
lc_5	Linear	1	n/a	n/a	n/a

The loss function of the critic is defined by $\frac{1}{m}\sum_{i=1}^{m} f_w(x^{(i)}) - \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))$, which can be broken down into two key components. The first part of the equation assesses the critic's evaluation on real tensors and is represented by $fw(R) = \frac{1}{m}\sum_{i=1}^{m} f_w(x^{(i)})$, where m corresponds to the number of real tensors within the dataset, $x^{(i)}$ signifies the i-th real tensor, and f_w denotes the function responsible for evaluating the critic's response to a given tensor. Conversely, the second part of the equation involves the evaluation of the critic on the generated tensors and is expressed as $fw(G) = \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))$, where m represents the number of generated tensors, and $z^{(i)}$ represents the i-th generated tensor.

Ultimately, the equation $fw(R) - fw(G)$quantifies the disparity between the assessments made by the critic on real and generated tensors. This value is instrumental in updating the weights of the generator and signifies the discrepancy in scores assigned by the critic to real and generated (fake) data, with the aim of minimizing this difference and enhancing the quality of the generated tensors.

Arjovsky et al. [7] identified a limitation within the WGAN algorithm, specifically concerning the utilization of weight clipping to enforce a Lipschitz constraint. It is observed that this approach is ineffective, as it poses challenges in achieving optimal training of the critic network. If the clipping parameter is too large, it prolongs the time required for the weights to reach their limit. Conversely, if the parameter is too small, issues such as vanishing gradients could arise.To address this limitation, Gulrajani et al. [9] introduced an enhancement to WGANs by incorporating a gradient penalty parameter to enforce the Lipschitz constraint. This enhancement contributed to maintaining stability during training. The gradient penalty is introduced to ensure that the critic network maintained a bounded gradient, thereby mitigating concerns related to exploding gradients and model degeneration. As a result of this improvement, modifications were made to the loss function of the critic in WGANs to include the gradient penalty term. The adjusted loss function for the critic is expressed by $fw(R) - fw(G) + \lambda \mathbb{E}_{\hat{x}\sim\mathbb{P}\hat{x}}[(|\nabla_{\hat{x}} fw(\hat{x})|_2 - 1)^2]$, where λ functions as the scaling factor for the penalty term, $\mathbb{E}_{\hat{x}\sim\mathbb{P}\hat{x}}$ represents the mathematical expectation over samples of $\hat{x}$, and $\hat{x}$ denotes a randomly sampled interpolation between pairs of real and fake samples. In this interpolation process, an initial random value, ϵ, is generated within the range [-1,1]. Subsequently, this value is multiplied by the real tensor and the residual value, $1 - \epsilon$, is calculated. The generated tensor is then multiplied by this residual value, and the two tensors are subsequently summed. Finally, $\mathbb{P}\hat{x}$ denotes the empirical distribution of $\hat{x}$, and $fw(\hat{x})$ represents the critic's output for a given sample $\hat{x}$. The operator $\nabla\hat{x}$ denotes the gradient of the function $fw(\hat{x})$ concerning $\hat{x}$, and $|\cdot|_2$ represents the Euclidean norm. Regarding hyperparameters, these encompass a latent dimension of 256, a tensor size of 64×64, 1149 epochs, 24 channels, and a batch size of 64. Furthermore, the hyperparameter λ is established at a value of 10.

2.7 AE Architecture

An AE represents a type of neural network model rooted in unsupervised learning, which is proficient at data compression (encoding) and subsequent decompression (decoding). Its primary objective is to replicate the input vector in the model's output, a process driven by the features learned during training.

This configuration adheres to the conventional AE setup, where an encoder precedes a decoder. In this context, the decoder takes the form of the generator within the pre-trained WGAN network with fixed weights. The data used for training this network corresponds to

the same dataset employed in training the WGAN network.

The primary aim of this stage is for the encoder to acquire the capacity to generate effective latent spaces for the recreation of healthy data. The loss function employed focuses on minimizing residual loss through mean squared error (MSE). It can be expressed as

$$\frac{1}{n} \parallel x^{(i)} - g_\theta(E(x^{(i)})) \parallel^2 + \frac{\kappa}{n_d} \parallel f(x^{(i)}) - f(g_\theta(E(x^{(i)}))) \parallel^2,$$

where $\parallel . \parallel^2$ denotes the sum of squared residuals for each value within the tensor, the term $f(.)$ represents the critic features from an intermediate layer, κ serves as a weighting factor, g_θ denotes the WGAN generator, n_d corresponds to the dimensionality of the intermediate feature representation, n stands for the number of coefficients within the tensor, and $x^{(i)}$ represents the i-th real training tensor. It is important to emphasize that as explained in subsection 2.6, the early stopping technique is also used in this stage of training to avoid overfitting.

Table 4 presents the configuration details of the encoder architecture. It consists of five layers: le_1, le_2, le_3, le_4, which are convolutional layers featuring leaky ReLU activation functions and a dropout rate of 0.25 for each. Furthermore, they incorporate batch normalization layers with identical output values. Lastly, the le_5 layer takes the form of a linear layer equipped with a hyperbolic tangent activation function, yielding an output dimension matching the latent space (256). Hyperparameters include a latent dimension set to 256, a tensor size of 64×64, 200 epochs, 24 channels, and a batch size of 64. The final hyperparameter, κ, is established at a value of 1. Finally, the comprehensive methodology can be summarized using the flow chart

Table 4: Encoder stage architecture in the WGAN.

Name	**Type**	**Output shape**	**Kernel**	**Stride**	**Padding**
le_1	Conv2d	[16,32,32]	3	2	1
le_2	Conv2d	[32,16,16]	3	2	1
le_3	Conv2d	[64,8,8]	3	2	1
le_4	Conv2d	[128,4,4]	3	2	1
le_5	Linear	256	n/a	n/a	n/a

presented in Figure 2.

2.8 Damage Detection Indicator

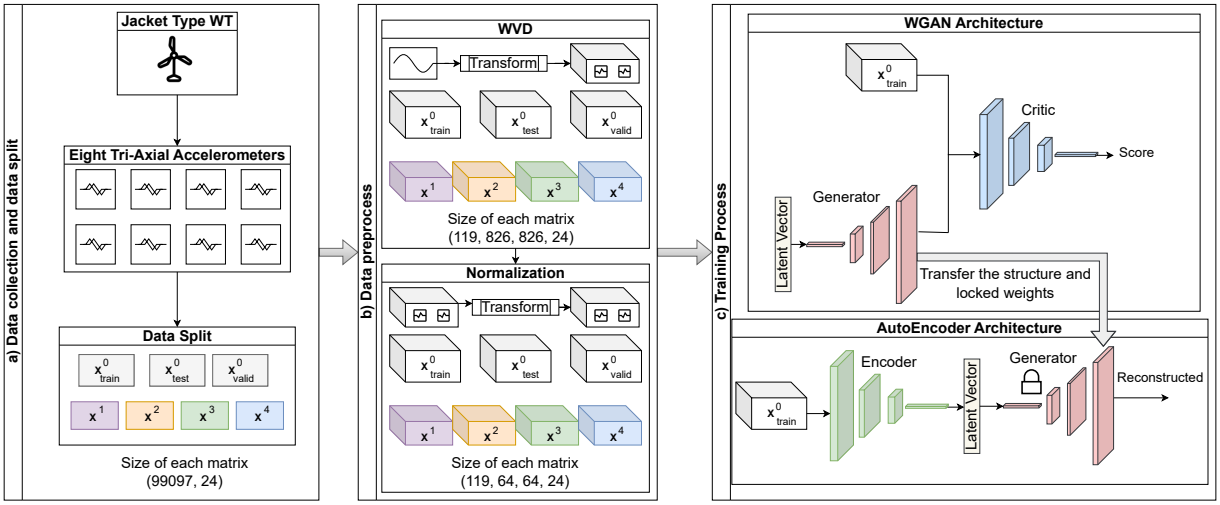
To evaluate the network's ability to detect anomalies, it's crucial to establish an indicator that, based on the outcomes from trained networks, can distinguish between healthy and unhealthy data. This indicator is referred to as the anomaly score and is defined as follows:

$$\alpha_s = \alpha_r(x^{(i)}) + \kappa\alpha_d(x^{(i)}), \tag{2}$$

where, $\alpha_r(x^{(i)}) = \frac{1}{n} \parallel x^{(i)} - g_\theta(E(x^{(i)})) \parallel^2$ represents the anomaly score derived from the difference between the values in the input tensor and the generated tensor, and $\alpha_d(x^{(i)}) = \frac{\kappa}{n_d} \parallel f(x^{(i)}) - f(g_\theta(E(x^{(i)}))) \parallel^2$ signifies the anomaly score obtained by subtracting the critical features of the input tensor from those of the generated tensor.

In cases of healthy data, anomaly scores tend to be low as the model excels at generating tensors resembling healthy ones. Consequently, differences between these tensors are minimal.

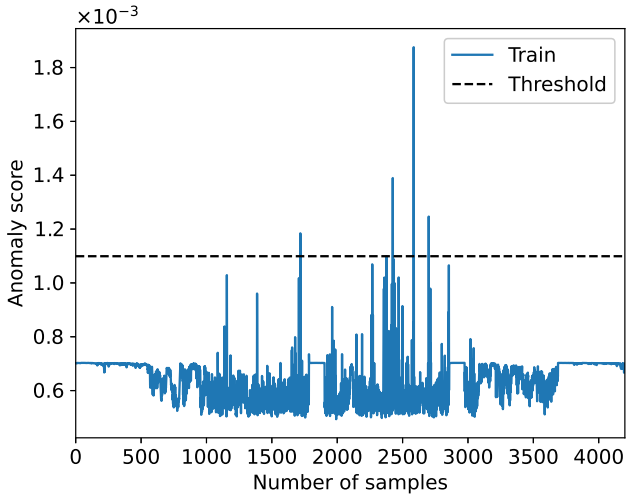
Figure 2: Flow chart of the methodological process for training the model.



However, with unhealthy data, the model still generates healthy data, leading to significantly increased disparities between these tensors and the actual data, resulting in higher anomaly scores.

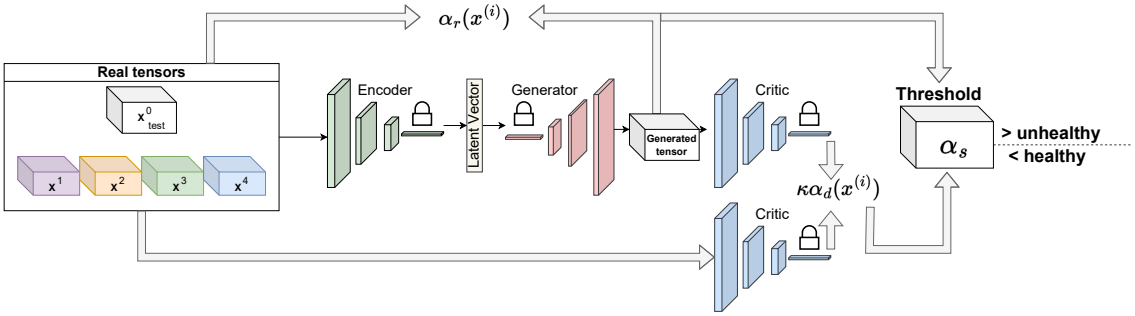
To establish a threshold, the aforementioned metric is employed to compute anomaly scores within the training data. Assuming a normal distribution for anomaly scores, the six-sigma rule, detailed in [10], is applied. As shown in Figure 3, the computed threshold value is $\mu + 6\sigma = 0.001099$, where μ and σ denote the mean and standard deviation of the anomaly score values, derived solely from the X^0_{train} data. Therefore, results that exceed this threshold indicate damaged state samples.

Figure 3: Threshold obtained with the six-sigma rule.



Finally, once the threshold for diagnosis has been obtained, the anomaly score is computed for the test set, which is composed of X^0_{test}, X^1, X^2, X^3, X^4. The flowchart of the proposed methodology is given in Figure 4.

Figure 4: Flowchart of the proposed methodology.



3 RESULTS

In this section, the results are presented, beginning with the training and validation stages of the WGAN and AE models. The times required for the training, validation, and inference processes of the experiments are detailed, providing a comprehensive view of the efficiency of the proposed models. Furthermore, the results of the model evaluation are presented in a test set composed of both healthy and unhealthy data, highlighting their ability to accurately and effectively discern between both categories.

Figure 5 (left) illustrates that the training loss of the critic in the WGAN network converges to zero, indicating the accurate reproduction of healthy tensors. The validation loss curve also approaches zero, which means a successful model generalization. The early stopping technique is applied, saving network weights at the epoch with minimal validation loss. In this experiment, epoch 49, marked by a vertical black line in Figure 5 (left), exhibits the best performance with a validation loss of -0.122767 and a training loss of -1.742411. The critic aims to maximize the gap between $P(g)$ and $P(r)$, so the loss curve of the critic near zero signifies the similarity of the generated tensors to the healthy ones. Figure 5 (right) displays the training and validation loss of the AE network's encoder stage, indicating a convergence to near-zero encoder loss. This convergence signifies the effectiveness of the encoder in generating latent vectors for constructing healthy tensors. Both validation and training loss curves closely match, reflecting successful model generalization. The early stopping technique is applied, preventing overfitting by storing the network's weights at the best epoch. Minimum losses occur around epoch 20, with values close to zero (0.000585), as shown by a vertical dashed black line in Figure 5 (right).

Table 5 presents information regarding the time required for training, validation, and inference processes. It consists of various metrics associated with the time taken for these tasks. The time is measured in hours, minutes, and seconds. The time required to process an entire experiment, which consists of 119 samples, is similar in all cases. This time becomes relevant when there is a need to infer a massive quantity of samples in the model.

Another interesting observation lies in the outcomes derived from bolts examined across various levels. For the purposes of the research, detection is carried out successfully, as can be seen in Figure 6 (left). The threshold, marked with a horizontal dashed black line, is significantly distant from the anomalies detected at various levels of loose bolts. Notably, the data associated with the healthy samples closely aligns with the designated threshold, yet consistently registers values beneath it across all instances 6 (right). A significant difference in amplitude range is observed between the unhealthy samples and the healthy samples. Greater stability with much smaller anomaly score amplitude ranges is typically exhibited by the healthy

Figure 5: Critic loss per epochs (left). Encoder loss per epoch (right).

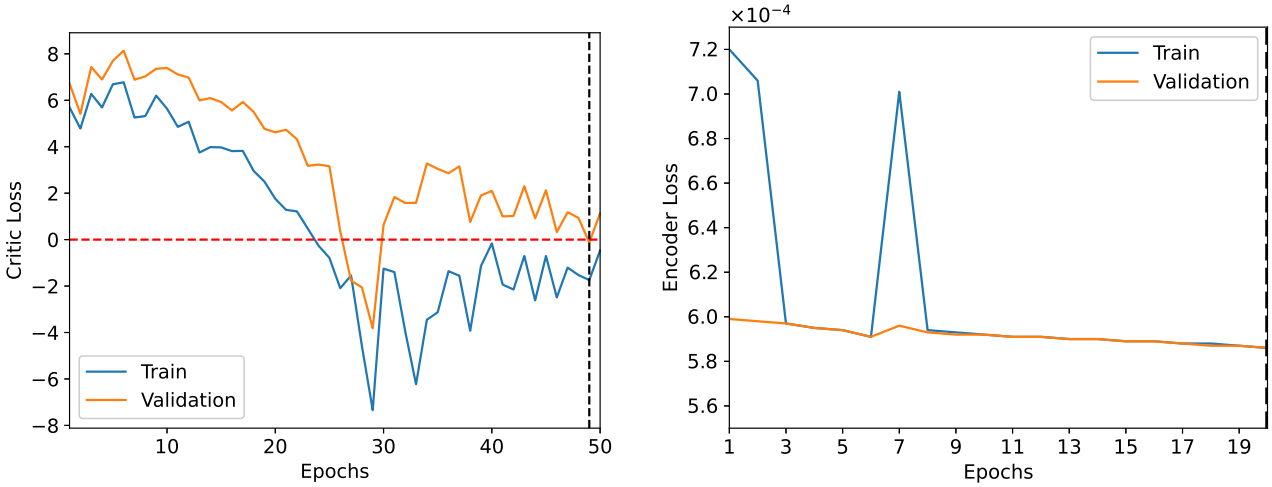


Table 5: Duration time (H:m.s) for training, validation, and inference.

	Epochs	**Total time**	**Time by epoch**	**Time by experiment**
WGAN train	50	07:19.90	00:08.80	00:00.24
WGAN validation	50	01:01.10	00:01.22	00:00.24
AE train	20	02:39.80	00:07.99	00:00.22
AE validation	20	00:22.20	00:01.11	00:00.22
Inference	-	00:21.22	-	00:00.25

samples. However, the results reveal an overlap in the characteristics of bolts at different levels, see Figure 6 (left), underscoring the challenge inherent in distinguishing them. This overlap suggests an opportunity to dig deeper with a thorough investigation, using statistical methods and machine learning. The goal of this new exploration is to uncover a clear and meaningful difference among the various levels. This will enhance the evaluation system's ability to tell them apart accurately.

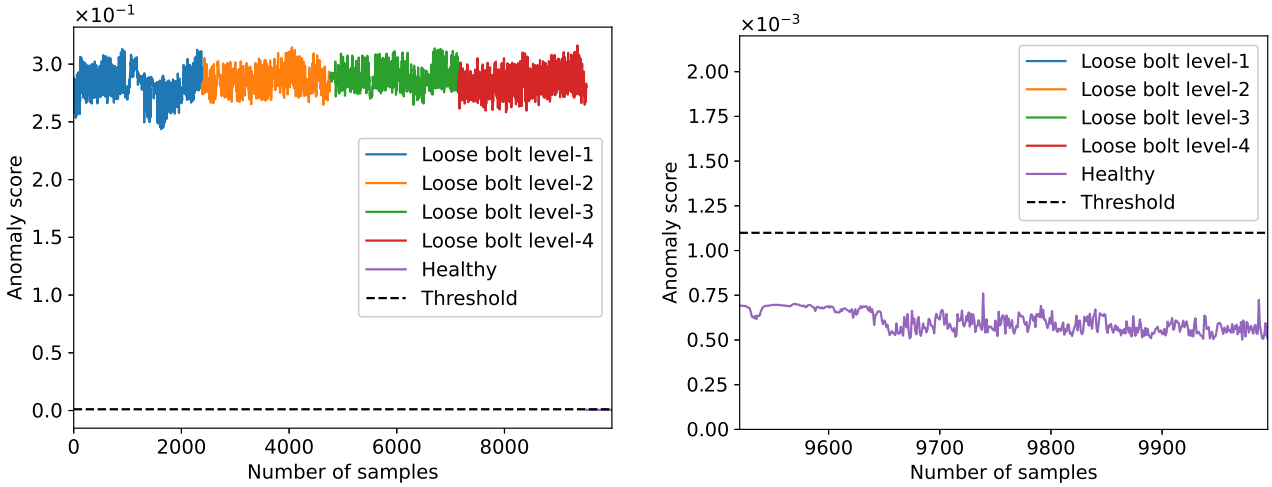
The data derived from experiments with loose bolts provides evidence that the neural network has a remarkable capacity to accurately discern between samples with loose bolts and healthy data, achieving a precision rate of 100%.

4 CONCLUSIONS

This study demonstrates the effectiveness of combining WGAN and AE networks for unsupervised anomaly detection in structures using accelerometer data. When applied to a jacket-type offshore WT structure, this methodology achieved precision 100% in differentiating cases of loose bolts from healthy cases, showcasing its potential as an SHM methodology.

Although these proof-of-concept results in a controlled environment are promising, their feasibility and impact in the real world require further validation. Additional data sets that encompass diverse structural conditions are needed to thoroughly evaluate the robustness of this approach. Furthermore, real-world testing under complex loading and environmental conditions will determine the viability of the methodology for SHM across various structures.

Figure 6: Results of early detection of loose bolts (left). Zoom-in healthy data samples (right).



ACKNOWLEDGMENTS

This work is partially funded by the Spanish Agencia Estatal de Investigación (AEI) - Ministerio de Economía, Industria y Competitividad (MINECO), and the Fondo Europeo de Desarrollo Regional (FEDER) through the research projects PID2021-122132OB-C21 and TED2021-129512B-I00; and by the Generalitat de Catalunya through the research project 2021-SGR-01044.

REFERENCES

[1] 2020 *Energy* **202** 117787 ISSN 0360-5442

[2] Vidal Y, Aquino G, Pozo F and Gutiérrez-Arias J E M 2020 *Sensors* **20** 1835

[3] Feijóo M d C, Zambrano Y, Vidal Y and Tutivén C 2021 *Sensors* **21** 3333

[4] Mao J, Wang H and Spencer Jr B F 2021 *Structural Health Monitoring* **20** 1609–1626

[5] Schlegl T, Seeböck P, Waldstein S M, Langs G and Schmidt-Erfurth U 2019 *Medical image analysis* **54** 30–44

[6] Puruncajas B, Vidal Y and Tutivén C 2020 *Sensors* **20** 3429

[7] Arjovsky M, Chintala S and Bottou L 2017 Wasserstein gan (*Preprint* `1701.07875`)

[8] Böhland M, Bruch R, Löffler K and Reischl M 2023 *Current Directions in Biomedical Engineering* **9** 467–470 URL `https://doi.org/10.1515/cdbme-2023-1117`

[9] Gulrajani I, Ahmed F, Arjovsky M, Dumoulin V and Courville A C 2017 *CoRR* **abs/1704.00028** (*Preprint* `1704.00028`)

[10] Pyzdek T and Keller P 2014 *Six sigma handbook* (McGraw-Hill Education)