IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 27 July 2023, accepted 9 August 2023, date of publication 21 August 2023, date of current version 28 August 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3306958

RESEARCH ARTICLE

On Anomaly-Aware Structural Health Monitoring at the Extreme Edge

DAVID ARNAIZ[1,2], EDUARD ALARCÓN[1,3], (Member, IEEE), FRANCESC MOLL[1], (Senior Member, IEEE), AND XAVIER VILAJOSANA[2,4], (Senior Member, IEEE)

[1]Department of Electronic Engineering, Universitat Politécnica de Catalunya (UPC), 08034 Barcelona, Spain
[2]Worldsensing, 08014 Barcelona, Spain
[3]NaNoNetworking Center in Catalonia, Universitat Politécnica de Catalunya (UPC), 08034 Barcelona, Spain
[4]Wireless Networks Research Laboratory (WiNe), Universitat Oberta de Catalunya (UOC), 08018 Barcelona, Spain

Corresponding author: David Arnaiz (david.arnaiz@upc.edu)

This work was supported in part by the Industrial Doctorate Plan of the Department of Research and Universities of the Generalitat de Catalunya. The work of David Arnaiz was supported by Agència de Gestió d'Ajuts Universitaris de Recerca under Grant AGAUR 2019 DI 075.

ABSTRACT Self-awareness has been successfully utilized to create adaptive behaviors in wireless sensor nodes. However, its adoption can be daunting in scenarios, such as structural health monitoring, where the monitored environment is too complex for it to be accurately modeled by a sensor node. This article addresses this challenge by proposing a novel and lightweight anomaly-aware monitoring method for structural health monitoring that can be directly executed by a sensor node. Instead of modeling the complete structure, the proposed anomaly-aware monitoring method uses the vibration measurements of the sensor node to identify local deviations in the dynamic response of the monitored structure. The self-awareness module can then use this information to guide the dynamic behavior of the sensor node, replacing more resource-intensive structural models. We use data from multiple public benchmark structures to evaluate different features and propose an unsupervised feature selection method. Additionally, we evaluate different anomaly detection algorithms comparing their ability to detect local structural damages, also taking into account their memory and energy cost. The proposed method has been implemented in a commercial sensor node, and deployed in a scaled structure where various damage scenarios were simulated to validate the proposed method, where it was able to successfully detect the presence of damages in over 88% of the cases. Finally, we showcase how the proposed method can enhance self-awareness through the use of a simulation, where the proposed monitoring method was able to extend the battery life of the sensor node by over 59%, without impacting the node's ability to swiftly detect damages in the structure.

INDEX TERMS Internet of Things (IoT), wireless sensor networks (WSN), anomaly detection, structural health monitoring, self-awareness, structural monitoring.

I. INTRODUCTION

Civil structures play a critical role in modern society, as many of us rely on bridges, tunnels, railways, and buildings, among others, for our everyday affairs. Consequently, correctly maintaining these structures is not only critical to prevent unplanned downtime, but also because failures in these structures may suppose a safety hazard for its users. Despite this, many structures are not correctly maintained or are even known to be in unhealthy states. The American Society of Civil Infrastructure (ASCE) Report Card from 2021 reported that 7.5% of the bridges in the US are considered structurally deficient and the US has a backlog of $ 125 billion in repairs [1]. Even in cases where the damaged bridges are identified, they are not correctly maintained because of the limited budget of the organizations responsible for these structures.

Reducing the maintenance cost of the structures requires shifting from reactive to proactive maintenance, detecting and repairing potential problems early before they turn into

The associate editor coordinating the review of this manuscript and approving it for publication was Renato Ferrero.

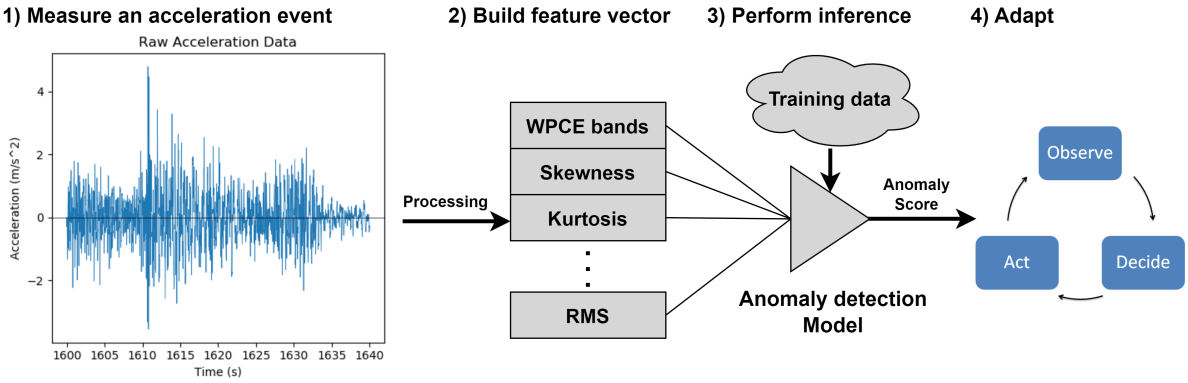


FIGURE 1. Overview of the proposed anomaly-aware monitoring method.

costly damages. To this end, in the past decade, significant effort has been devoted to the development of Wireless Sensor Networks (WSN) [2], [3] and Structural Health Monitoring (SHM) [4], [5]. WSNs facilitate the collection of data from the monitored structure in a reliable, autonomous, simple to install, and inexpensive way [6]; while SHM provides the analytic capabilities to model the state or *'health'* of the structure using the collected data. The combination of these two technologies enables the evaluation of the state of the monitored structure in real-time, minimizing the need for manual inspections, and reducing the repair costs by detecting failures early or even before they actually take place, increasing the safety and lifetime of the structure [7].

Despite the progress made in these two technologies, their adaption to civil structures is still limited. One of the biggest challenges for the widespread adoption of wireless sensors is their limited resource availability, particularly their battery life. This becomes particularly problematic in data-intensive applications such as vibration monitoring, typically used for SHM [8]. The problem of how to optimally allocate resources in wireless sensors has attracted significant attention from the research community since the inception of this technology. Nevertheless, as the optimal configuration of the sensor nodes is highly dependent on the environment being monitored, sensor nodes need to dynamically adjust their behavior based on their operating conditions. Over the years countless dynamic energy management methods have been proposed to address this problem [9], [10], and have proven to be able to reduce the energy consumption of the sensor nodes while maintaining their Quality of Service (QoS). The main drawback of these methods is the loss of predictability on the sensor node's behavior, since it becomes environment dependent. Maintaining predictability is paramount in safety-critical applications to prevent the loss of data. In recent years the notion of *Self-Awareness* [11] has been used to address this problem, by allowing the node to adjust to the monitoring environment while being able to retain some predictability in the form of operational goals.

Introducing self-awareness at the sensor node level for SHM is particularly challenging due to the size and complexity of the monitored structures. Modeling the health condition of the structure requires collecting data from multiple distributed sensor nodes, and aggregating it using resource-intensive models. A single sensor node only has access to its local environment and lacks the resources to run such models, thus, is unable to model the monitored structure in a reliable way. This hinders the implementation of self-awareness at the sensor node level, as it removes the ability to react to changes in the monitored environment. Alternatively, the cloud application can relay this information back to the nodes [12]. However, this approach is only adequate to model slow changes in the condition of the structure, not for real-time decision-making, since there will be some delay from when the node performs the measurement and when it gets an update on the state of the structure from the cloud.

A vibration-based SHM application collects the vibration data from multiple points of the structure to analyze the dynamic response of the structure. If the structure gets damaged, its dynamic response varies, allowing the SHM application to detect and identify such damage. An individual sensor node does not have access to this information, as it can only access its local measurements. However, it can generate a model of its local vibration patterns, and identify any deviation that may take place. Any local damage will introduce changes in the vibration patterns measured by the sensor node, allowing it to detect local damages. The node may not have enough information to assess the source of the damage, or even if the observed deviation has been caused by the presence of damages, but it has enough information to detect that some previously unseen event is taking place and has to be studied in detail.

Even in cases where the deviations in the observed vibration patterns are not introduced by the presence of damages (for example if the structure was loaded by a previously unseen wind direction) collecting detailed information about these anomalous events is more useful to the SHM application than the data collected when the structure is in a known state,

as the latter is already accounted for by the model. Therefore, the sensor node should increase its monitoring efforts when the data is considered to be more anomalous, and it can compensate for this increment by reducing the monitoring efforts when the data is closer to nominal.

The proposed anomaly-aware monitoring method uses an anomaly detection algorithm to model the local dynamic response of the healthy structure using the vibration data collected by the sensor node. Then, it uses this model to assign an *anomaly* or *novelty* score to the monitoring data, indicating the extent to which it has deviated from the nominal, or healthy, state. This score is utilized by the self-awareness module of the sensor node to manage the resource allocation based on the perceived usefulness of the measured data.

This article, evaluates the use of a lightweight anomaly detection algorithm at the sensor node level for SHM applications to compute the anomaly score of the data, and evaluates potential use to guide the node's behavior. Figure 1 provides an overview of the proposed anomaly-aware monitoring method, which will be thoroughly discussed and explained in detail. The main contributions can be summarized as follows.

- We propose a novel, lightweight anomaly-aware monitoring method for SHM that identifies deviations in the dynamic behavior of the monitored structure to dynamically adjust the node's behavior.
- We use public SHM benchmarks to propose an unsupervised feature selection method based on the achieved feature performance. Additionally, we compare common anomaly detection algorithms, evaluating their performance, computational and memory cost.
- We validate the proposed monitoring method with a commercial node and a scaled structure, simulating damage conditions on the structure.
- Finally, we showcase how the proposed monitoring method can be used to enhance self-awareness at the sensor node level through a simulation.

The rest of this paper is structured as follows. Section II, reviews the background and related works on self-awareness, damage detection in SHM, and ML for embedded systems. Section III continues by presenting a high-level description of the proposed anomaly-aware monitoring method. Section IV introduces the feature selection method and evaluates common, lightweight, anomaly detection algorithms. The setup and results of the validation test are commented on in Section V. Section VI showcases how the anomaly-aware monitoring method can be used to implement self-awareness at the sensor node level through the use of a simulation. Finally, the conclusions and future work are summarized in Section VII.

II. BACKGROUND AND RELATED WORK

This work was motivated by the need of introducing self-awareness at the sensor node level in the context of SHM, where modeling the environment is not viable. The notion of self-awareness is introduced in subsection II-A. Then, subsection II-B evaluates the existing examples of self-awareness in the context of wireless sensor nodes. Subsection II-C brings the focus back to SHM by reviewing the relevant works in damage detection methods. Finally, subsection II-D evaluates the relevant works in anomaly detection methods for resource-constrained systems.

A. SELF-AWARENESS

Self-awareness [13], [14], changes the paradigm of how systems are managed in dynamic environments. Computer systems are expected to operate reliably and efficiently under a wide range of operating conditions, and thus, their behavior needs to be adjusted accordingly. Traditionally, to adjust the system behavior, the system designers had to analyze all the system behavior under all possible operating conditions and define the low-level adaptive policies that the system should follow based on the perceived environment. This approach is problematic as the operating conditions and their effects on the system may not be a priori known, and defining the low-level policies necessitates a deep understanding of the system's inner workings and its operating environment. Self-awareness provides a promising solution, by giving devices the ability to self-manage. Instead of following a set of predefined low-level policies, a self-aware system is able to autonomously define and perform adaptive actions at runtime based on its high-level operational goals, which can be defined without knowledge about the system's inner workings or its operating environment.

A computing system can be considered as being self-aware if it is capable of monitoring itself and its environment, and autonomously adjusting its behavior at runtime in order to achieve its goals in an efficient way. To achieve self-awareness, a self-aware system needs sensors to collect information about its internal parameters (e.g. CPU load, power consumption) and environmental parameters (e.g. atmospheric temperature, risk factor), which is then used to generate and maintain models. These models allow the system to observe its current state, its goals, its potential actions, and its environmental conditions. Additionally, a self-aware system needs to be able to use its models to predict the effect of its current operating conditions on its ability to achieve its goals and decide the best adaptive actions to take in response, taking into account the predicted effect of these actions on itself and its environment. Finally, a self-aware system needs to be able to execute the defined adaptive plan. The main traits of a self-aware system are summarized in Figure 2. Generally, all this functionality is organized in the form of an Observe-Decide-Act (ODA) loop [15].

B. SELF-AWARENESS IN WIRELESS SENSOR NODES

The self-managing nature of self-awareness has attracted the attention of researchers in different research fields such as resource-constrained cyber-physical systems [16], system on chip [17], computer networks [18], among many others.

FIGURE 2. Main traits of a self-aware system.

In this section, we review the relevant work in self-aware wireless sensor nodes.

The principles of self-awareness have been applied to different aspects of sensor nodes. In the network, Zhuang et al. [19] applied the principles of self-awareness for congestion control in multi-hop wireless sensor networks. Each sensor in the network is able to monitor, and affect its transmission bandwidth using lossy compression, at the cost of introducing distortion to the data. The congestion control method proposed by Zhuang et al. allowed the sensor nodes in the network to autonomously allocate the distortion budget between them to prevent network congestion, while minimizing the introduced distortion.

Regarding data processing in sensor nodes, Forooghifar et al. [20] applied self-awareness to an epileptic seizure detection classifier. The proposed approach uses multiple hierarchical classifiers with increasing complexity. Self-awareness is used to select the appropriate classifier for the data to maintain the required accuracy level, while minimizing the energy consumption of the classification. The self-aware classifier proposed by Forooghifar et al. only takes into account one parameter, the expected accuracy of the classifiers for the given data, to guide the selection. Other self-aware patient monitoring systems are able to improve the performance by also taking into account the risk factor of the patient [21], [22], although this requires having access to information from additional sensors and increased processing, which is not viable to implement in a singular sensor node.

Self-awareness has also been applied to the data acquisition function of wireless sensor nodes. Arnaiz et al. [23] proposed a data acquisition method using the principles of self-awareness. Given a target battery lifetime for the sensor node, the proposed method is able to dynamically adjust the sampling period to achieve the target battery life while maximizing the Quality of Service (QoS) of the wireless sensor. The QoS of the sensor node is improved by increasing the sampling rate when the signal is considered more *"relevant"* and compensating for this increment during periods when the signal is deemed less important. To model the data relevance, the authors use a synthetic metric (i.e. the mean deviance of the measured data). The Mean deviance does not model the state of the environment, but provides statistical information about the collected data. Using this metric, the proposed method was able to outperform equivalent naive sampling methods, thus, showing that modeling the collected data is also advantageous.

The self-aware-based methods presented in this section have one main difference from the more conventional dynamic energy management methods in that they are able to not only adjust to the operating conditions, but are also able to comply with their given operational targets. In all the presented examples, the sensor nodes are able to monitor at least one internal parameter and have some kind of model of their environmental context. Modeling the complete environment can be complex and resource intensive in applications such as SHM, hindering the use of self-awareness in these cases. Instead of modeling the complete structure, the anomaly-aware monitoring method presented in this paper models the vibration measurements of the sensor node, allowing the sensor to react to the changes in the measured data.

C. DAMAGE DETECTION IN SHM

SHM is the area of research focused on answering the question of what is the condition (or *health*) of a given structure through non-destructive means. It is distinguishable from the more traditional structure maintenance processes in that SHM is performed automatically on near real-time bases, instead of relying on periodic inspections. One of the main aspects of SHM is structural damage identification. In this context, structural damage can be defined as "changes introduced into a system that adversely affect its current or future performance" [24].

There are a myriad of methods for SHM, however, they are usually classified as model-driven, if they use an analytical model of the monitored structure; or as data-driven, if they use statistical analysis of the data collected from the structure, to detect, locate and identify the presence of damages [25]. Typically, analytical models are Finite Element Models (FEM) of the monitored structure. Simulating FEM models is a resource-intensive task, which is usually performed by cloud servers. Consequently, in this subsection, we will focus on reviewing relevant lightweight data-driven methods.

Data-driven models can be further classified depending on the tools used to analyze the data, as time domain such as Autoregressive Moving Average (ARMA) [26] and Bayesian probabilistic methods [27], or as frequency domain such as Wavelet Packet Decomposition (WPD) or Fourier-based tools. Han et al. [28] used WPD to compute a damage index based on the spectral energy distribution with promising results. WPD is commonly used over the traditional Fourier transform for SHM as it has better properties for non-stationary signals, which are common in this domain. These

data-driven methods perform statistical analysis of the data to detect when the structure deviates from its nominal operation. These models are significantly more lightweight than other approaches and have shown interesting results. However, due to their simplicity, these models are commonly less robust to changes in environmental conditions and sensor noise.

More advanced ML methods have also been used for SHM, Malekloo et al. [25] reviewed the existing literature about the use of ML for SHM. One challenge of using ML models for SHM is that typically, there is only data available of the structure in its healthy state, and thus, novelty/anomaly detection models are more widely used than supervised learning models. As an example, Sarmadi and Karamodin [29] proposed an adaptive Mahalanobis squared distance and one-class KNN-based algorithm for SHM, trained exclusively from healthy data, which was able to detect damages under varying environmental conditions. ML models are generally more computationally expensive and require extensive training. Nevertheless, they have been shown to be able to operate under varying environmental conditions.

Most of the efforts in SHM focus on the accuracy of the proposed methods and their ability to operate under varying temperature conditions. However, little effort has been done into evaluating the cost of running such models in terms of computational power and memory usage, since such models are not aimed at resource-constrained systems. This article evaluates different lightweight anomaly detection algorithms in terms of performance, computational cost, and memory usage to understand the tradeoffs and decide the best candidate to be used in resource-constrained wireless sensors.

D. ANOMALY DETECTION METHODS FOR RESOURCE-CONSTRAINED EMBEDDED SYSTEMS

The advancements in microelectronic manufacturing, have allowed for the development of more capable and power-efficient embedded systems. The performance improvements of embedded systems, combined with the recent progress in optimizing ML models, have made it possible to deploy these models directly on edge devices. This area of research is known as TinyML [30]. Processing part of the data locally, instead of outsourcing the computation to the cloud has some advantages: it reduces the latency, improves privacy and security, reduces bandwidth, and in wireless sensors allows reducing the energy consumption required to transmit the data. Nevertheless, local processing also has some drawbacks, as it increases the processor and memory usage, and due to its lightweight nature, it will be generally less accurate than its resource-intensive counterpart.

For the purpose of this paper, we focus on lightweight anomaly detection methods. Domingues et al. [31] made a comparison between the fourteen most prevalent anomaly detection methods using multiple datasets, evaluating their accuracy, robustness, processing time and memory usage. As a result of their evaluation, the authors favored the isolation forest (or I-Forest) algorithm for its scalability and accuracy with large datasets; and One-Class Support Vector Machine (OCSVM) for its performance with smaller datasets. Similarly, McKinnon et al. [32] compared the accuracy between Elliptic Envelope (EE), I-Forest, and OCSVM; using a dataset extracted from a wind turbine gearbox data. The test showed that the I-Forest and OCSVM algorithms obtained similar results and outperformed the EE algorithm. This shows a clear preference for these two algorithms I-Forest and OCSVM.

Similarly, this article evaluates four different anomaly detection algorithms, I-Forest, OCSVM, Gaussian Mixture Model with Mahalanobis Distance (GMM-MD), and Z-score. Unlike the existing literature, it uses data from four public SHM datasets, evaluating the model's ability to identify light and severe damages in the structure. Additionally, during the evaluations, the inference is executed directly in a sensor node to accurately measure the memory usage and computation cost of each algorithm.

III. ANOMALY-AWARE MONITORING

Vibration monitoring is the most widely used method for SHM [33], as it is able to detect superficial and internal damages. In wireless sensor nodes, vibrations are commonly measured using Micro-Electro-Mechanical System (MEMS) accelerometers, due to their small size and low energy consumption [34].

The typical operation of a wireless sensor for vibration monitoring is shown in Figure 3 (a). To save energy, sensor nodes remain in sleep mode most of the time, only waking up periodically to start a monitoring event before returning to sleep mode. Monitoring events can be started periodically with a predefined periodicity, or be triggered by external factors. In vibration monitoring, monitoring events are commonly triggered based on the vibration level of the structure. Commonly, in vibration monitoring, the monitoring events are triggered when the vibration intensity surpasses the threshold value. Having a high enough vibration level is critical for the accuracy of the measurements as it is directly related to the Signal-to-Noise Ratio (SNR) of the measurements. Therefore, sensor nodes designed for vibration monitoring remain in sleep mode maintaining the MEMS accelerometer active, waiting for the vibration level of the signal to exceed a predefined threshold to start a new monitoring event.

Unlike static measurements, such as temperature, humidity, or inclination, where a single data point contains enough information to characterize the monitored parameter; the dynamic response of the structure is characterized by the temporal changes in the accelerometer data over a period of time. Therefore, in vibration monitoring, sensor nodes have to acquire a series of acceleration measurements for a window of time. The duration of the acceleration measurement window can range between a few seconds to several minutes, sampled at a rate of 0.1 to 4kHz depending on the structure being monitored and the available memory in the sensor node. Transmitting the complete window of data using

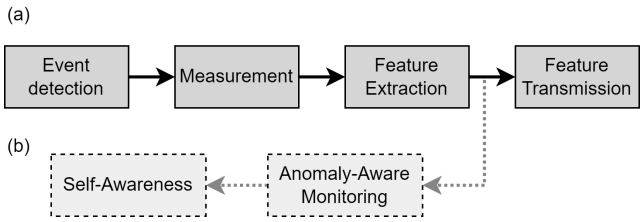


FIGURE 3. Vibration monitoring flow, (a) standard and (b) additions for anomaly-aware monitoring.

the node's wireless interface is not usually a viable option due to the bandwidth restrictions of the most commonly used Low Power Wide Area Networks (LPWANs), such as LoRaWAN [35]. Instead, sensor nodes have to preprocess the accelerometer data locally to extract a set of relevant features that reflect the main characteristics of the collected signal in a compact form.

With the proposed anomaly-aware monitoring method, when the nodes are first deployed in the structure, they have no prior knowledge or model of the dynamic response of the structure. Sensor nodes start in *learning mode*, performing the standard processing flow from Figure 3 (a), collecting a window of accelerometer data, computing a set of features, and transmitting those features using their wireless interface. Sensor nodes remain in learning mode while collecting enough data to train the anomaly detection model. It is important that during this phase, the sensor nodes collect data from the structure in different operating conditions (e.g. different load distributions, or excitation sources) and environmental conditions (e.g. different temperatures or humidity levels), so as to correctly model the behavior of the model.

Once a sensor node has gathered enough data, the data collected by the sensor node is used to generate a model of the structure using an anomaly detection algorithm. The generated model is transmitted to the sensor node, ending the training phase and switching the node to *inference mode*. In inference mode, the sensor node performs the additional operations from Figure 3 (b). During each monitoring event, the anomaly detection model processes the features extracted in the current monitoring event, calculating an *anomaly score*, which is then used to guide the behavior of the node.

The anomaly score value provides information on how different are the current feature values from the ones used to train the model. If the dynamic response of the system has not changed, the anomaly scores obtained will remain low. If the structure's response changes due to the presence of damages or is operating under previously unseen conditions, the anomaly scores will increase. It should be noted that the actual anomaly score value does not provide any information on its own, as the actual score depends on the model and the training data used. However, two measurements can be compared using their anomaly score value to determine the most anomalous one. If the anomaly score value obtained during a monitoring event is high, it may be an indication

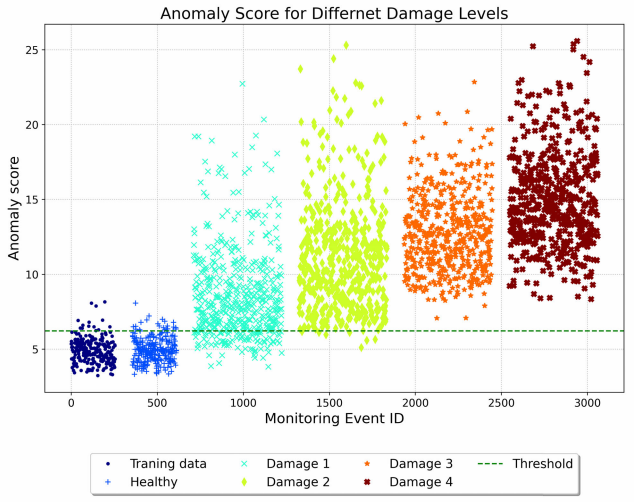


FIGURE 4. Anomaly score for increasingly more severe damage test cases.

that the structure is damaged or has changed in some way, and thus, it should be prioritized over others.

The anomaly score value can be used by the self-awareness module to implement different adaptive policies. One such policy is to only transmit the data with the highest anomaly scores. By only transmitting the anomalous data, the sensor node can significantly increase its battery life, without impacting its ability to detect if the structure is damaged, as the presence of damages will increase the anomaly score of the measurement, so they would be transmitted. Figure 4 shows the anomaly scores obtained with the Qatar University Grandstand Simulator benchmark data [36], [37], [38] (see Section IV-A4) with increasingly more severe damage conditions. Each marker in the graph shows the score of a single monitoring event; the horizontal green dashed line shows the anomaly detection threshold set to the 90th percentile of the training data. As can be seen, as the damage in the structure increases, so does the calculated anomaly score, for the two most severe damage test cases all of the monitored events are correctly identified. In this case, the anomaly-aware monitoring method would be able to save energy by only sending 10% of the data, without having a significant impact on the ability of the SHM application to detect damages.

Structures progressively degrade over time, until they eventually get damaged. To minimize the maintenance cost and improve the safety of the structure is important to identify and repair these damages early—before they grow into more severe and costly structural failures—. However, when the structure is only slightly damaged, it will not experience large changes in its dynamic response, and thus, it will not cause large increments in the anomaly score value calculated by the sensor node. In these cases, the anomaly detection model may not correctly label all the monitoring events as damaged due to the weak change in the structure's response. Nevertheless, even if not all the cases are correctly labeled, the overall scores calculated by the node while the structure is slightly

damaged will be larger than the scores obtained when the structure was completely healthy. In such cases, analyzing the temporal evolution of the anomaly score can provide a useful insight to identify slight, but gradual changes in the monitored structure. These changes can be characterized using the moving percentile of the anomaly scores, which is calculated using Algorithm 1. Where α controls the weight of previous values, and $0 < \alpha < 1$; p is the percentile value; *score* is the current score value; m_0 is the moving percentile value of the previous iteration; and m is the updated moving percentile value.

An example of how the moving percentile of the anomaly score value provides information about the state of the monitored structure is shown in Figure 5. The data in the graph is from the SMC Cable Stayed Bridge benchmark [39] (see Section IV-A2). The graph shows the evolution of the anomaly score as the bridge degrades. The data used to train the anomaly detection model is shown in green, the data used to evaluate the model is shown in orange, and the horizontal dashed green line represents the anomaly detection threshold set as the 90th percentile of the training data. Two separate damages take place in the benchmark structure while it was being monitored. The date when these damages took place is not known as they were identified on a later date, during a routine inspection, however, the anomaly scores clearly show two points in time where the anomaly score increased. These dates are marked by the vertical red dashed lines. In this dataset, the effect of the damages is weak, particularly for the first event, and thus, most of the monitoring events are not correctly classified as anomalies. However, it can be seen that the moving 90th percentile of the scores does react to the damage, thus allowing the sensor node to identify the two damage cases.

Overall, the proposed anomaly score monitoring method presented in this paper extends the conventional sensor nodes' vibration monitoring flow, allowing them to calculate an anomaly score of their measurement as an indication of how much has the structure deviated from its nominal state. The anomaly score of a single monitoring event can be used to detect severe damages, enabling the SHM application to generate early responses, or it can use the moving percentile of anomaly scores to identify the effect of slight damages over a period of time. While the presence of damage produces changes in the dynamic response of the structure, resulting in larger anomaly scores, it is important to note that an increase in the anomaly score does not necessarily indicate the presence of damage. The sensor node does not have a model of the structure to evaluate the possible cause of the damage and discard any changes due to the structure operating in previously unseen conditions. Nonetheless, increasing the monitoring effort in such conditions also provides valuable insights for the SHM application. Therefore, the anomaly-aware monitoring method presented in this paper is intended as an efficient data collection method rather than a stand-alone alarm generation system.

Algorithm 1 Moving Percentile Calculation

Parameters: *initialized*, α

procedure MovingPercentile(p, *score*, m_0)
  $q \leftarrow (p/100)$
  if *initialized* **then**
    initialized $\leftarrow$ *True*
    $m \leftarrow$ *score*
  else if *score* $< m_0$ **then**
    $m \leftarrow m_0 - \alpha/q$
  else if *score* $> m_0$ **then**
    $m \leftarrow m_0 + \alpha/(1-q)$
  else
    $m \leftarrow m_0$
  return m

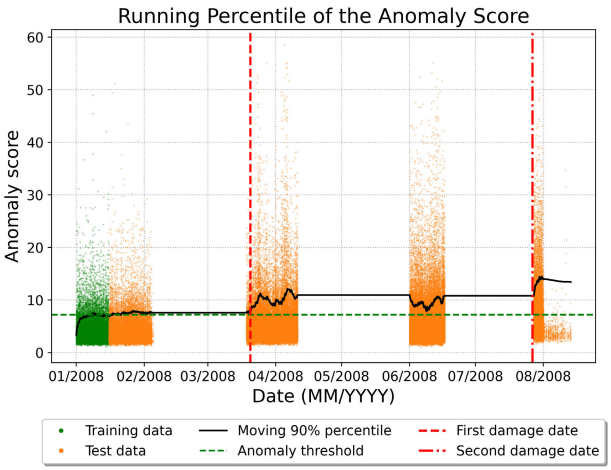


FIGURE 5. Temporal evolution of the moving percentile of the anomaly score.

IV. BENCHMARK EVALUATIONS

We assess various features, feature selection methods, and anomaly detection algorithms to determine the optimal strategy to implement the anomaly-aware monitoring method presented in this paper. For the evaluation, we utilized data from four distinct public SHM benchmarks, which were collected from different structures experiencing different damage conditions. The varied range of conditions of the different benchmarks allows us to identify how the proposed solution will behave in a generic SHM application, and thus, provide a general solution, instead of one focused on a given structure or damage condition.

A. SHM BENCHMARKS

The benchmarks used for the evaluation are publicly available benchmarks commonly used in the literature to evaluate SHM damage detection methods. In this subsection, we provide a brief overview of the different benchmark structures, the damage conditions evaluated, and the data collection settings for each of the benchmarks. The benchmarks consist of data collected from multiple sensors distributed across

the structure and include various test cases with different levels of structural damage. To simplify the evaluation process, from each benchmark we only considered data from one of the sensors, and only from three different test cases. The first test case we selected was used as a reference for the healthy, *undamaged* state of the structure. The second test case was selected to represent the structure with severe damage, referred to as *damage condition 1*, which gives an example where the damage can be easily detected. Finally, the last test case, which is referred to as *damage test case 2*, was selected to represent a case where the structure is slightly damaged, making this test the hardest to identify.

The data of the benchmarks are presented as a continuous time series of the accelerometer measurements sampled over the duration of each test case. To generate multiple monitoring events from these data, we divided the data into windows of 1000 acceleration samples. Each window represents the data collected in a single monitoring event of a sensor node. The length of 1000 samples is mainly limited by the memory capacity of the commercial sensor node used to evaluate the algorithms. The data on some benchmarks is very limited, so we applied some overlapping between consecutive windows to increase the number of windows available for the test. The degree of overlapping was adjusted based on the data available on each benchmark.

To prepare the data before the evaluation we performed some preprocessing. Each data window was preprocessed independently as if it corresponded to a monitoring event from a sensor node. The preprocessing consisted of first a first-order detrending, so as to remove the offset and drift that was observed in some of the benchmark data. After the detrending, we normalized the energy of the vibration signals, by dividing the acceleration samples by their Root Mean Square (RMS) value. The energy normalization process eliminates the temporal variability in the vibration level, making sure that changes in the vibration source between the undamaged and damaged test cases are not used to identify these cases. In addition to this, some benchmarks required additional preprocessing. In these cases, benchmark-specific preprocessing is introduced in conjunction with the benchmark description in the remaining part of this subsection.

1) BENCHMARK 1: IASC-ASCE SHM TASK GROUP BENCHMARK STRUCTURE

On August 2002 the IASC-ASCE SHM Task Group performed experimental studies on a test structure simulating damages by removing bracing or loosening bolts [40]. The test structure is a four-story, two-bay by two-bay steel frame scaled-model structure with a scaling factor of 1 to 3. The structure was monitored using fifteen accelerometers, three per floor including the base of the structure. The accelerometers were placed on the center column, east wall, and west wall of each floor of the structure. The test structure was studied under nine different test conditions, which are detailed in Table 1. In each of the test cases, the response of the structure was studied under four different sources of excitation: ambient, shaker performing a sine sweep, shaker with random vibration, and hammer.

For the simulation tests presented in this paper, only the data collected when the structure was loaded with the shaker in random mode were used, so as to make sure there is enough vibration energy in the data to extract useful information, and because of its similarity with the sources used for other benchmarks. From the fifteen accelerometers used to collect the benchmark data, only the data from the accelerometer on the fourth floor, the east face was used, since it is located directly where most of the damage is introduced in the test cases. From all the available test cases, case 1 is used as the reference, undamaged case; case 9 is used as the severe damage condition; and case 5 is used as the slightly damaged case. In test case 5, since all the east face braces are removed, the response of the structure becomes similar to the undamaged case, making it more complex to detect than the rest of the test cases. The acceleration measurements of the benchmark were sampled at 250 Hz for at least 2.5 minutes per test case. Because of the little amount of data available for some of the test cases, for this benchmark, the window length was reduced to 500 samples and set the overlapping between windows to 60%.

2) BENCHMARK 2: SMC CABLE-STAYED BRIDGE

The Center of Structural Monitoring and Control (SMC) at the Harbin Institute of Technology [39] published the data collected by the SHM system from an actual cable-stayed bridge. The bridge is a three-span concrete bridge with a total length of 510 m and a width of 11 m. The SHM monitoring data is composed of the measurements of fifteen accelerometer sensors collected with a sampling rate of 100 Hz to monitor the dynamic response of the bridge, in conjunction with temperature and wind sensors to monitor the environmental conditions of the bridge. The accelerometer data is available since the 1st of January 2008. Then, during an inspection carried out on August 2008, two damaged patterns were detected, and the bridge was eventually closed to traffic. It is believed that the bridge gradually degraded due to overloading, but the exact date when the two damages occurred is unknown.

The resonance frequencies of the bridge are at very low frequencies, so to remove unnecessary data,the original data was filtered using a fourth-order low pass Butterworth filter with a cut-off frequency of 25 Hz, and we downsampled it by a factor of two. When the data was recorded, the bridge was not subjected to any active excitation sources, as the other benchmarks. Instead, the vibration of the bridge is caused by external sources, such as traffic and wind. Therefore, any acceleration window with an RMS value below $0.02\,m/s^2$ was discarded, so as to ensure the energy of the used data was sufficiently high. The utilized threshold was defined so that no background vibration was considered while keeping all the traffic-induced events. For this benchmark,

TABLE 1. Test cases for the IASC-ASCE SHM Group experimental tests.

Case	Description
1	Undamaged structure.
2	The first-floor brace on the south corner on the east face was removed.
3	Case 2 + the top floor brace on the south corner of the east face was removed.
4	Case 3 + the remaining braces on the south corner of the east side were removed.
5	Case 4 + all remaining braces on the east face were removed.
6	Case 5 + the second-floor braces on the north face were removed.
7	Case 6 + the remaining braces from all faces of the structure were removed.
8	Case 7 + the bolts on both ends of the beams on the first and second floors on the north side of the east face were loosened.
9	Case 8 + the bolts on both ends of the remaining beams on the north side of the east face were loosened.

TABLE 2. Selected test cases for the SMC bridge benchmark.

Case	Dates (YYYY-MM-DD HH)
Undamaged	2008-01-01 10 - 2008-01-10 22
slight damage	2008-04-03 10 - 2008-04-13 22
severe damage	2008-07-27 10 - 2008-07-31 22

since there was more than enough data for the tests, no overlapping was applied.

The monitored bridge in this benchmark gradually degraded, thus, there are no clearly distinctive test cases. Huang et al. [41] analyzed the data from the benchmark using a Kalman filter-based method and observed that larger variations in the dynamic response of the bridge took place around May and larger changes towards the end of June. The dates used to determine the test conditions used for the evaluations are detailed in Table 2. Only the data recorded using sensor 1 was used for the evaluations, as it is located the closest to where both damaged patterns were identified during the August inspection.

3) BENCHMARK 3: K.U. LEUVEN Z24 BRIDGE

The Z24 bridge was a Swiss concrete two-cell box-girder bridge with a main span of 30 m and two side spans of 14 m. In 1998, prior to its demolition, the KU Leuven Structural Mechanics Section introduced artificial damages to the bridge, while monitoring its dynamic response of the bridge under different test conditions. The instrumentation and the test conditions of the bridge are explained in the original publication by De Roeck et al. [42], the follow-up publications [43], [44], and the K.U. Leuven webpage [45]. The damage tests were divided into two parts, during the first part simulated damages in the pier settlement, simulated by lowering the pier; then, during the second part the pier was left in its original position and different damage conditions were simulated.

For the evaluations presented in this paper, only the pier settlement cases were used, as they provide progressively more severe test cases. The test cases for the first part of the Z24 benchmark can be seen in Table 3. Case 2 is used as the undamaged condition, to avoid mistakenly identifying

TABLE 3. Initial test cases of the Z24 bridge benchmark.

Case	Description
1	Undamaged bridge.
2	Pier settlement system was installed.
3	Lowering of pier, 20mm.
4	Lowering of pier, 40mm.
5	Lowering of pier, 80mm.
6	Lowering of pier, 95mm.
7	Lifting of pier, tilt of foundation.

the pier settlement equipment as a damage, case 3 is used as the slight damage test condition, and case 6 as the severe damage condition. The KU Leuven Structural Mechanics Section monitored the dynamic response of the bridge using 62 accelerometers under ambient and forced vibration. Only the data measured using sensor 209 was used for the test, as it experienced the most change when the damages were applied, and only the measurements from the forced vibration tests, as they have more energy. The original data from the benchmark was collected with a sampling rate of 100 Hz, using an antialiasing filter with a cut-off frequency of 30Hz. To generate enough data windows for the evaluations, the overlapping between windows was fixed to 40%.

4) BENCHMARK 4: QATAR UNIVERSITY GRANDSTAND SIMULATOR

The Qatar University Grandstand Simulator (QUGS) structure [36], [37], [38] is a 4.2 m x 4.2 m hot-rolled steel frame, consisting of 30 joints. The frame was designed to carry a total of 30 spectators. The vibrations on the structure were monitored by an accelerometer located at each joint of the frame, with a sampling rate of 1024 Hz, and each event was recorded for 256 s. During the tests, the structure was excited using a shaker, which generated vibration at random frequencies. The QUGS structure was tested under 31 different test cases, one undamaged, and the rest with the bolts loosened in each of the joints. The test cases were repeated two times, generating two different datasets, which are available at Qatar University website [46].

The evaluation only considers the data from the sensor located at the first joint. The undamaged test case was used as

the reference case, the test case where the damage is located at the first joint as the severe damage test scenario, and the test case where the damage is located at the joint located at the opposite corner of the structure as the slight damage test case. Since there are two different datasets available for the same structure, the data from the first dataset was used to train the model, and data from the second dataset for the evaluation test cases.

B. FEATURE SELECTION

The acceleration time series, which records the dynamic response of the structure, is the basis for building the feature vector in the second step of the proposed anomaly-aware monitoring method (as shown in Figure 1). The feature vector is a compressed representation of the time series data, emphasizing its significant attributes. This subsection evaluates different features and introduces the proposed unsupervised feature selection method for SHM.

1) FEATURE EVALUATION METHOD

The feature selection manages the tradeoff between the complexity of the anomaly detection algorithm and its performance. Not all features are equally sensitive to the presence of damages, and furthermore, some features are sensitive only to a certain type of damages. To obtain a better compromise in this tradeoff it is important to select only those features that will produce the best results. Ideally, a feature should have similar values for data points in the same class and values as different as possible for other classes. In supervised learning applications, Fisher score [47] is the most common method to evaluate the performance of a given feature. In SHM applications the data from the damaged structure is generally not available and there is no prior knowledge about the failure mode. Thus, without access to the failure data, there is no clear way of evaluating the performance of a given feature. Instead, the proposed unsupervised feature selection method evaluates the statistical performance of the features in the benchmark structures to evaluate which features provide the best performance for a wide range of structures and failure modes.

We evaluated the features by computing the Fisher score for all the features using the data from the different benchmarks. Each benchmark has two damaged scenarios and an undamaged one. Since the goal of the damage detection algorithm is to identify any potential damage, we combined the data from the two damage scenarios. The Fisher score measures the sensitivity of a feature to the presence of damages. However, it is possible that two different features are highly correlated and have similar sensitivity to the damages, making them redundant. The redundancy between features is commonly evaluated using the Pearson Correlation Coefficient (PCC), which measures the linear dependence between two random variables. The PCC gives a value between -1 and 1. A PCC of zero indicates that the two variables are not linearly dependent, and a value close to 1 or -1 means they are linearly dependent (inversely dependent in the case of -1). The PCC can be calculated using Equation 1, with X and Y being the two features being evaluated, μ_I being the mean value of feature I, and σ_I being the standard deviation of feature I.

$$\rho_{XY} = \frac{E\left[(X-\mu_X)\,(Y-\mu_Y)\right]}{\sigma_X \sigma_Y} \tag{1}$$

Two features may be correlated when the structure is undamaged, but become uncorrelated in the presence of damage. To consider a feature redundant, it needs to be correlated before, and after the damage appeared. Consequently, the correlation coefficient needs to be aggregated using Equation 2, where ρ^0_{XY} is the PCC of features X and Y on the undamaged case, and ρ^1_{XY} is the PCC of features X and Y in the combined damaged test cases.

$$\widehat{\rho_{XY}} = \left|\rho^0_{XY}\right| + \left|\rho^1_{XY}\right| \tag{2}$$

The aggregated PCC values are used to determine which features are redundant. For the evaluations presented in this paper, we defined a threshold of 1.4 for the combined PCC. If the PCC of two features is higher than the threshold value, the one with the lowest Fisher score is discarded. Once all the redundant features have been discarded, the optimal features are the ones with the highest Fisher scores.

2) WAVELET PACKET COMPONENT ENERGY

The frequency response of the monitored structure is commonly used to evaluate the health condition of the monitored structure, since it is generally sensitive to the presence of damages. In the proposed feature selection method, we evaluate the use of WPD to extract this information. For the fast discrete wavelet transform, the data is passed through a pair of quadrature mirror filters, one acting as a low-pass (*scaling*) filter and the other as a high-pass (*wavelet*) filter. The low-pass filtered data is referred to as approximation (A) coefficients and the high-pass filter data is referred to as detail (D) coefficients. The filtered signals are then decimated by a factor of two, so the total number of samples in the detail and approximation coefficients match the number of samples in the original signal. A new decomposition level can be obtained by applying the same process to the approximation coefficients. As the signal is decomposed it will generate a partial decomposition tree as shown in Figure 6 (a).

WPD [48] is a generalization of the wavelet transform where the detail coefficients are also processed in subsequent decompositions, generating the full decomposition tree as shown in Figure 6 (b). Effectively each decomposition splits the signal into 2^j frequency bands, with j being the number of decompositions. As the number of decompositions increases, the frequency bands get narrower, so the frequency resolution increases; but the number of samples per band decreases, reducing the temporal resolution.

Sun and Chang [49] demonstrated using a numerical simulation of a three-span bridge that the Wavelet Packet Component Energy (WPCE) is a robust indicator even in the

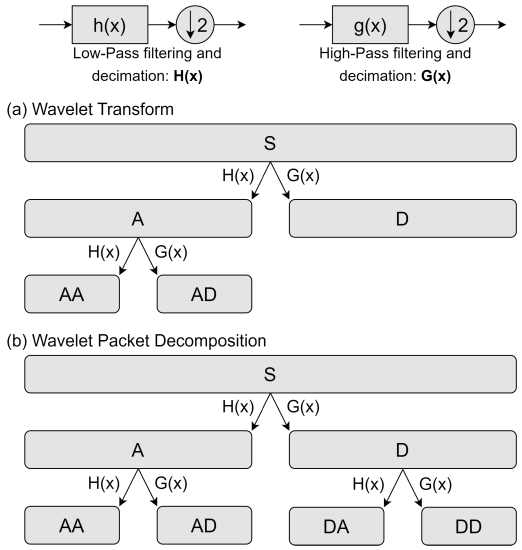


FIGURE 6. Comparison between (a) wavelet tree decomposition and (b) wavelet packet tree decomposition, showing two decomposition levels.

presence of noise. WPCE has been successfully tested in other use cases, such as beam structures [28], ASCE benchmark structure [50], and sub-sea pipeline bedding condition assessment [51], to name a few; showing promising results in all cases.

If the mother wavelet used for the decomposition is orthogonal, the energy of the signal is the sum of the energy from all the frequency bands [49], as shown by Equation 3.

$$E_f = \sum_{i=1}^{j^2} E_{f_j^i} \tag{3}$$

where E_f is the total energy of the signal, and $E_{f_j^i}$ is wavelet packet component energy stored in the i th frequency band. The energy stored in the component signal $f_j^i(t)$ is calculated using Equation 4,

$$E_{f_j^i} = \int_{-\infty}^{+\infty} f_j^i(t)dt \tag{4}$$

The WPCE gives a vector with the energy stored in each frequency band after the decomposition. This vector can be normalized to obtain the percentage of energy stored in each frequency band, thus, removing the dependence on the vibration level when the signal was collected. The normalization is done by dividing the energy in each frequency band $E_{f_j^i}$ by the total energy E_j, using Equation 5.

$$\hat{E_{f_j^i}} = \frac{E_{f_j^i}}{\sum_{i=1}^{j^2} E_{f_j^i}} \tag{5}$$

There is a wide variety of orthogonal wavelets reported in the literature. The most common family of wavelets for SHM is the Daubechies (db), shown in Figure 7. The Daubechies wavelet family is commonly referred to as db$N$, where the

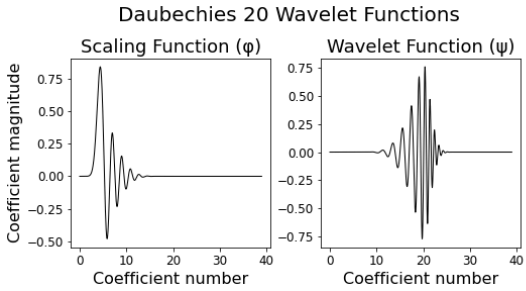


FIGURE 7. Daubechies with 20 vanishing moments (db20) wavelet scaling and wavelet functions.

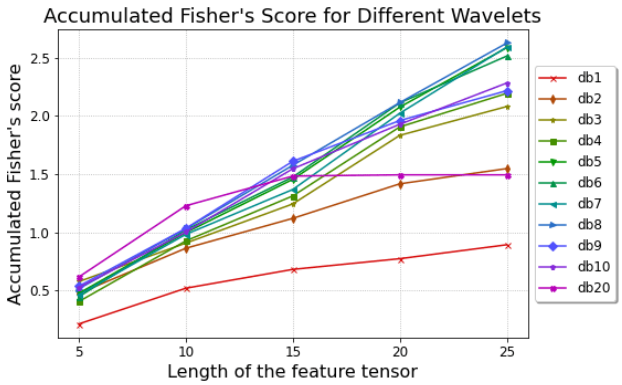


FIGURE 8. Accumulated Fisher's scores for different mother wavelets.

N expresses the order or the number of vanishing moments of the wavelet. The wavelet's order also defines the number of coefficients in the low-pass and high-pass filters used to decompose the signal, which is $2N$. The order of the wavelet is a key parameter that needs to be adjusted. As the order of the wavelet increases, the leakage between frequency bands is reduced. However, it reduces the maximum number of decompositions that may be done for a given signal length, and consequently, the frequency resolution that can be achieved.

The performance of different wavelet orders for different sizes of feature vectors was evaluated using the procedure from Section IV-B1, using the WPCE bands as features. Since each structure has unique resonance frequencies, each benchmark was evaluated separately and the results were obtained by taking the geometric mean of the individual results. The average accumulated Fisher scores for the evaluated wavelets and for different lengths of the feature vector are shown in Figure 8. The optimal wavelet order depends on the dimension of the feature vector. However, the wavelets of orders 5 to 8 perform best during most of the range, with the db8 wavelet obtaining the best overall performance. Consequently, the db8 wavelet was used as the mother wavelet for the remaining tests presented in this paper.

3) FEATURE SELECTION METHOD

Apart from the normalized WPCE, there are countless other features that can be extracted from the acceleration data.

TABLE 4. Features considered for the proposed feature selection method.

Feature name	feature name
Maximum	Crest factor
Minimum	Kurtosis factor
Mean	*Pulse factor*
Peak-to-peak	Margin factor
Mean of absolute	*Upper control limit*
Standard deviation	*Lower control limit*
Skewness	Form factor
Kurtosis	Inter quartile range
Root Mean Square (RMS)	Mean absolute deviation

TABLE 5. Relative accumulated Fisher score obtained with the unsupervised WPCE selection method, with respect to the supervised method for the different benchmarks.

No. Feats	Benchmarks 1	2	3	4	Avg
5	73.14%	22.06%	21.27%	48.95%	36.00%
10	93.12%	31.69%	29.47%	68.78%	49.27%
15	79.62%	51.00%	37.16%	68.41%	56.68%
20	90.82%	62.70%	54.64%	78.26%	70.25%
25	91.28%	71.49%	56.25%	78.81%	73.34%

For example, some features measure the variability of the signal (standard deviation), its energy (RMS), or the shape of its probability density function (skewness, kurtosis). The features taken into account by the feature selection method are detailed in Table 4. It should be noted that some of the features in the table are affected by the detrending and energy normalization preprocessing performed on the benchmark data, nevertheless, they will still be listed in italics for completion.

Measurements from other sensors can also provide useful information that can be used as features. One clear example of this is the temperature, which is known to have an impact on the dynamic response of the system [52]. However, these data are not available for all the benchmarks, and thus, they were not taken into consideration during the valuation.

Scoring the features using the procedure from Section IV-B1, it can be seen that the features with the highest scores for all benchmarks are the WPCE bands. Nevertheless, the actual bands with the highest score depend on the structure being monitored.

The presence of damages in the structure changes its frequency response, changing how the energy is distributed. There are two possible effects: 1) a new resonance peak appears or disappears; 2) the energy shifts from a set of bands to a different set of bands. In the first case, even if the band where the new resonance peak changed is not being monitored, the normalized energy in the rest of the bands will be affected. The second case may be more problematic as if the energy changes from one band to another, and none of them is being monitored, the damage will not cause any noticeable effect. In this case, it is more likely that the energy shifts from a band that already had energy when the structure was healthy. Therefore, an intuitive method to select the WPCE bands is to select the bands with more energy when the structure is healthy.

On the other hand, from the common features only *skewness*, *kurtossis* and *mean absolute deviation*, remained after the correlation filter. Based on the obtained Fisher scores, these features are only preferable over the WPCE peaks once the feature vector already contains the main peaks, making them only useful in cases where the feature vector is large. The reduced memory availability in wireless sensor nodes limits the maximum number of features, thus, making these features less preferable.

Table 5 shows the relative accumulated score obtained using the accumulated Fisher score to select the optimal features (i.e. the supervised feature selection method), and the unsupervised feature selection method of selecting the WPCE bands with more energy during the training period. In Table 5 it can be seen that as the number of selected bands increases, the difference between the unsupervised and the supervised feature selection method decreases. Overall, the result shows that selecting the WPCE based on their undamaged energy is a good method if the number of bands is large.

During the learning process, when the wireless sensor nodes are first deployed in learning mode, the WPCE bands with more energy are still not known. The sensor node computes the complete WPCE vector and transmits the information of the bands with the highest energy, thus saving energy compared to sending the complete WPCE vector. However, Due to the variability in the structure's response, there is a chance that one of the peak WPCE bands is not transmitted, as another band may have achieved more energy during this monitoring event. These transmissions cannot be taken into account as they have at least one band missing, so the training phase needs to be extended to compensate for this. To avoid this effect and still save energy, the number of transmitted bands can be set higher than the maximum length of the feature vector, and adjusted based on the observed variability in the WPCE bands.

C. ANOMALY DETECTION ALGORITHMS

Anomaly detection is the third step of the proposed anomaly-aware monitoring method, as shown in Figure 1. During this step, the feature vector is processed to determine if the behavior of the structure has deviated from its nominal state. These deviations can indicate the presence of damages in the structure and are detected using anomaly detection algorithms, which are trained using only data from the healthy structure.

There are countless anomaly detection algorithms reported in the literature, each with its unique characteristics and applications. In our evaluation, we focus on the performance as well as the memory and computation efficiency of these algorithms, which are essential for their implementation in wireless sensor nodes. We assessed four algo-

rithms: OCSVM, I-Forest, Mahalanobis Distance with Gaussian Mixture Model (MD-GMM), and Z-score. OCSVM and I-Forest are the most widely used anomaly detection algorithms and have been shown to outperform other algorithms in a wide range of applications [31]. More focused on SHM, the Mahalanobis distance metric has obtained promising results [29]. We paired the Mahalanobis distance metric with a GMM, which defines the clusters from which to measure the distance. Lastly, the Z-score algorithm is particularly useful in resource-constrained systems due to its memory and computational efficiency.

1) ISOLATION FOREST (I-FOREST)

I-Forest [53] identifies outliers in the data based on how easy they are to separate, or *isolate*, from the rest of the data. An I-forest model is an ensemble of isolation trees, which are a special case of binary classification trees. Isolation trees define a random threshold value for one of the features to split the data into two groups. Each of those groups can be subdivided by defining another threshold value for the same or a different feature. This is repeated until each data point in the training is isolated from the rest, or until the maximum tree depth is reached. Outliers are different from the nominal data, and thus, they require fewer splits, or tree depth, to be isolated. The I-forest algorithm uses the average tree depth to calculate the anomaly score.

The number of trees in the ensemble is the main hyperparameter for the I-forest algorithm. Adding more trees to the ensemble increases the performance of the model—as more results are averaged—however, it also increases the computational and memory cost of the model.

2) ONE-CLASS SUPPORT VECTOR MACHINE (OCSVM)

The OCSVM algorithm is a special case of Support Vector Machine (SVM), which is trained using only data from the positive case (i.e. the undamaged case for SHM). The basic principle of SVM is to locate the hyperplane in the feature space that best separates the data from the two classes. In cases where the data cannot be linearly separated, the data is transformed into a higher dimensional space using a kernel function. The OCSVM model, as defined by Schölkopf et al. [54] and the variant evaluated in this paper, is trained to find the hyperplane that separates the training data from the origin while maximizing the distance to the origin.

There are different kernel functions that may be used to transform the data in SVMs, the most common ones are linear, polynomial, Radial Basis Function (RBF), and sigmoid kernels. Some of these kernels introduce additional hyperparameters, such as the kernel coefficient (*gamma*) for polynomial, sigmoid and RBF kernels; or the independent coefficient (*coef0*) required by the polynomial and sigmoid kernels.

3) GAUSSIAN MIXTURE MODEL WITH MAHALANOBIS DISTANCE (GMM-MD)

The GMM [55] is a probabilistic clustering algorithm that assumes all the data is generated from a mixture of k normal distributions. The GMM also provides the means for calculating the parameters of those distributions, such as their cluster centers. Anomalous data will be distributed far from the center of Gaussian distributions. Mahalanobis distance [56] provides a measure of distance between a point and the center of a normal distribution. The Mahalanobis distance is given by Equation 6. $\vec{\mu}$ is the mean vector of the cluster, S^{-1} is the inverse of the covariance matrix for the cluster with size $N \times N$ with N being the length of the feature vector, and $\vec{x}$ is the data point to be evaluated. In essence, the Mahalanobis distance measures the number of standard deviations between the current data point and the center of the distribution in the feature space.

$$d_M\left(\vec{\boldsymbol{x}}, Q\right) = \sqrt{\left(\vec{\boldsymbol{x}} - \vec{\boldsymbol{\mu}}\right)^T S^{-1}\left(\vec{\boldsymbol{x}} - \vec{\boldsymbol{\mu}}\right)} \tag{6}$$

4) Z-SCORE

The concept behind the Z-score anomaly detection model is similar to the GMM-MD model, but instead of defining the clusters and distance in the complete feature space, each feature is treated independently. Removing the dimensions significantly simplifies its processing as the covariance matrix gets replaced by the standard deviation of each feature. The anomaly score is given by Equation 7, where μ_f if the mean of feature f, σ_f is the variance of feature f, x_f is the feature value for the evaluated data point, and N is the length of the feature vector.

$$z\left(\vec{\boldsymbol{x}}\right) = \sum_{f=0}^{N} \frac{\left|x_f - \mu_f\right|}{\sigma_f} \tag{7}$$

D. HYPERPARAMETERS

To correctly compare the different anomaly detection models they need to have their hyperparameters optimally adjusted. A summary of the hyperparameters from all the models evaluated in this paper is shown in Table 6, along with the tested range for each parameter. As can be seen, particularly for the OCSVM, the total search space is considerably large. Before the evaluation tests, a set of coarse-grain simulations was performed using the benchmark data to reduce the search space.

Initial tests done on the I-Forest model show that when the number of estimators is low (<100), the performance of the model is significantly affected and has a large dependence on the stochastic nature of the training process (i.e. the feature and value used to split the data at each node), which cannot be realistically tested in an unsupervised way. For a large number of estimators, the memory required for the model proved to be too large to run on the Sensor node, thus, the maximum number of trees was limited to 500.

TABLE 6. Summary of the hyperparameters of the anomaly detection models.

Algorithm	Hyperparameter	Range
I-Forest	No. estimators	1 - 1000
OCSVM	Kernel	[linear, polynomial, RBF, sigmoid]
	Gamma	0.001 - 1000
	Degree [*1]	2 - 10
	Coef0 [*2]	-100 - 100
GMM-MD	No. clusters	1 - 25
Z-Score	N/A	N/A

[*1] Only for polynomial kernel
[*2] Only for polynomial and sigma kernel

For OCSVM the initial tests show that the Coef0 and Gamma parameters had little effect on the overall performance of the model, as long as they are close to the center of the range. The effect of the extreme values on the performance of the OCSVM highly depended on the stochastic nature of the test, helping to improve or significantly reduce the performance of the model. Nevertheless, as the models are trained in an unsupervised way, the variability in the performance is undesirable. Consequently, we fixed the values used for the gamma and coef0 parameters to the default values used by the Scikit-learn library [57]. The default gamma value is given by Equation 8, where N is the number of features and σ^2 is the variance of the training data. The default coef0 value is zero. In addition to this, the degree of the polynomial kernel did not show any noticeable improvements for large degree values, and consequently, we limited the maximum degree of the polynomial kernel to 5.

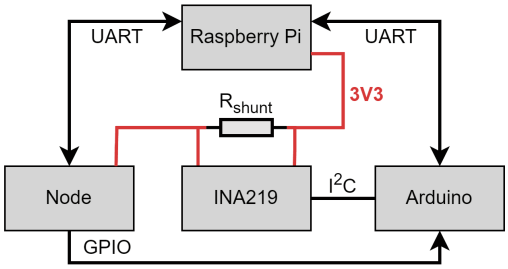
$$\gamma_{op} = \frac{1}{N\sigma^2} \tag{8}$$

Similarly, the coarse grain simulation tests showed no considerable improvement when the number of clusters defined for the GMM-MD algorithm was high, and thus, we limited the maximum number of clusters to 3.

E. EVALUATION SETUP

The evaluation setup is shown in Figure 9, (a) shows an interconnection diagram of the setup, and (b) a photograph of the actual setup. The main simulation script is executed in a Raspberry Pi, which iterates over the different benchmarks, anomaly detection algorithms, and their respective hyperparameters. For each evaluation, the Raspberry Pi was responsible for performing the feature selection process, training the anomaly detection models, and splitting the benchmark data into preprocessed accelerometer data windows. The feature extraction from the accelerometer data windows and the inference were executed in a slightly modified commercial sensor node. The commercial node used for the evaluation

(a) Interconnection diagram of the setup for the node evaluations.



(b) Labeled photo of the setup for the node evaluations.

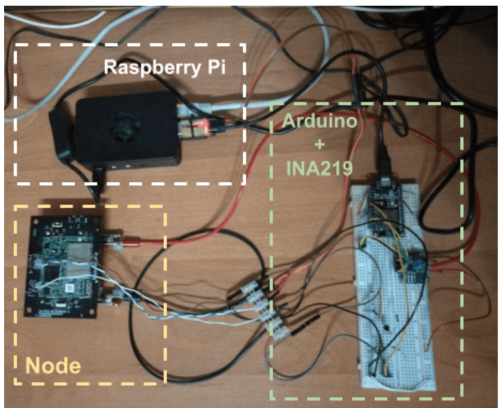


FIGURE 9. Setup for the evaluations in the sensor node (a) interconnection diagram of the setup, (b) labeled photo of the setup.

was a wireless tilt sensor manufactured by Worldsensing [58], which has an EFM32GG12B410F1024, ARM Cortex-M4 based microcontroller from Silicon Labs [59]. Executing the feature extraction and inference tasks on the node provides a more realistic evaluation environment, and also allows us to precisely monitor the energy consumption and memory overhead of the evaluated algorithms.

The power consumption of the sensor node was monitored using an INA219 power monitor IC [60], read by an Arduino board connected to the Raspberry Pi. A GPIO from the sensor node was used to indicate to the Arduino board the beginning and end of the inference, so only the inference power is recorded. In some cases, the inference time was too short to be accurately measured with this setup. To avoid this, the sensor node was programmed to perform each inference 200 times, making sure that the Arduino could accurately measure it.

In the proposed anomaly-aware monitoring method, the anomaly score threshold used to decide if a monitoring event is considered anomalous or not is dynamically adjusted by the self-awareness module depending on the needs at any given point in time. For the evaluation tests, the anomaly detection threshold was set to 90% of the anomaly score values obtained for the training data. The data from the undamaged test case was split into two equal halves, with one half used for training and the other half for the evaluation process. From the training data, two-thirds were used to train the damage detection model, and the remaining third to calculate

the anomaly detection threshold. To prevent sequential data from being concentrated in a single group, the data from the undamaged test case was randomly distributed across the three groups.

The stochastic nature of the evaluation process generates some variability in the results. For example, the data points used to train or evaluate the model may impact its overall performance; or the random splits generated by the I-forest algorithm will also affect the accuracy of the model. To eliminate this variability, each individual test was repeated 500 times, changing the stochastic parameters for each repetition. We averaged the results from the 500 repetitions to generate the final metrics of each evaluation. As the anomaly detection models will be trained while only having access to the healthy structure, removing the possibility to test the trained model before it is deployed; it is critical that the algorithm is robust, so that the obtained metrics have very little variability. To check the variability of the metrics obtained for each test, the standard deviation of each metric from the 500 repetitions was also recorded.

F. SIMULATION RESULTS AND CONCLUSIONS

1) MEMORY USAGE

Sensor nodes have very limited memory, which has to be shared between the ML model and the rest of the application. The model size obtained with the anomaly detection algorithms is a key parameter that determines if a model can be ported or not to a device. Figure 10, shows the average size in bytes of the models obtained from the evaluated algorithms across all benchmarks in the Y-axis, and the length of the feature vector in the X-axis. The legend shows the tested algorithms followed by their hyperparameter. For I-forest, the hyperparameter is the number of trees in the ensemble; for OCSVM the hyperparameter is the kernel, and for the polynomial kernel the order of the polynomial is shown between brackets; lastly, for the GMM-MD the hyperparameter is the number of clusters. The Z-score algorithm does not have hyperparameters, and thus, nothing is added to the legend.

I-forest is, with a difference, the algorithm with the highest memory requirements, remaining above $10kB$. The actual model size for the I-forest algorithm depends on how easy it is to isolate the data, so it changes from benchmark to benchmark. This variability in the model size is highly undesirable as it means that some memory buffer has to be reserved in case the trained model is large, or the maximum tree depth has to be limited which will impact its performance. The number of estimators also has a considerable impact on its memory requirements, since the parameters of each tree need to be stored independently. An interesting result about the I-forest algorithm is that the size of the feature vector has very little effect on the final model size, and thus, this algorithm may be comparable with the rest, in terms of memory usage, if the feature vector is increased, which matches the findings by Domingues et al. [31].

The next algorithm in terms of model size is the GMM-MD. Most of the memory of the model is used to store the inverse covariance matrix and mean vector of each cluster, which requires $m(n^2 + n)$ elements for n features and m clusters. The size of the OCSVM models is the second lowest and increases linearly with the number of features. The algorithm with the lowest memory cost is the Z-score, which just requires storing the standard deviation and mean value of each feature, resulting in $2n$ elements for n features. For 25 features, the GMM-MD algorithm with just one cluster requires approximately twice the memory of the OCSVM model with linear kernel, ten times more memory than the Z-score model, and four times less memory than the I-Forest algorithm with 100 trees.

2) INFERENCE CHARGE

The energy cost of performing the inferences is also a critical aspect to decide the optimal algorithm to implement in a sensor node, as it will affect the battery lifetime of the sensor node. The average battery charge required to perform a single inference is shown in Figure 11. The top image shows the charge consumption measured for all the evaluated algorithms, and the bottom image focuses on the algorithms with an inference charge consumption below $10nAh$.

As with the memory, the I-forest algorithm has the largest energy cost, more so as the number of trees in the ensemble increases. The energy cost of the I-forest algorithm does not only depend on the number of trees but also on the average tree depth required to isolate the data. Therefore, the battery charge consumption of the i-Forest model is highly dependent on the processed data. The main benefit of the i-Forest algorithm is that its energy cost slightly decreases with the length of the feature vector since with more features the data becomes easier to isolate. If the trend continues, the I-forest algorithms would be an interesting approach if the length of the feature vector is increased.

The next algorithm in terms of charge usage is the GMM-MD algorithm. The GMM-MD algorithm's complexity is $O(mn^2)$ for m clusters and n features, which limits the scalability of this algorithm. Nevertheless, for the evaluated feature vector lengths the inference charge consumption of the GMM-MD algorithm with just one cluster remains below $10nAh$.

For the OCSVM, the RBF kernel had the highest energy cost, with an inference charge consumption for 25 features above $20nAh$, while the rest of the kernels had a charge consumption below $4nAh$ for 25 features. The rest of the kernels had similar charge consumption, slightly higher for the kernels with higher orders. Due to its simplicity, the Z-score algorithm had the lowest inference charge consumption, remaining below $1nAh$ for 25 features.

The execution time of the anomaly detection algorithm is also a parameter of interest to compare the evaluated algorithms. The execution time and the consumed charge are related by Equation 9, where Q is the inference charge; I is the current consumption of the node while performing the

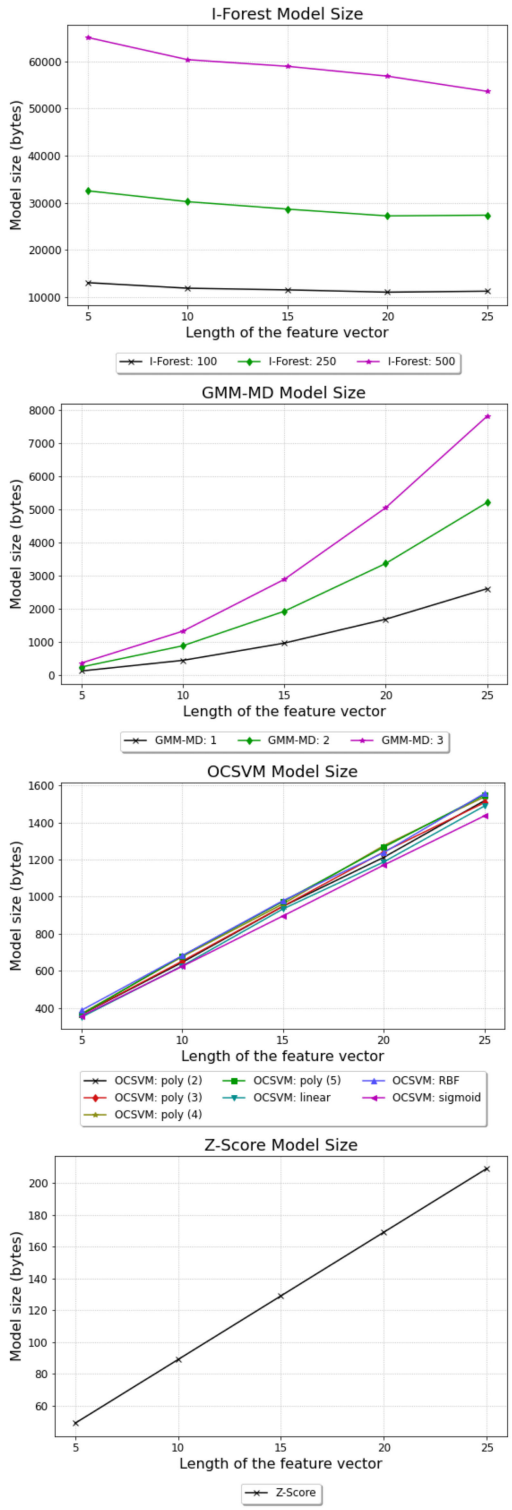


FIGURE 10. Memory usage of the anomaly detection models.

inference, with for the evaluation node is 22.3mA; and t is the execution time.

$$Q = It \tag{9}$$

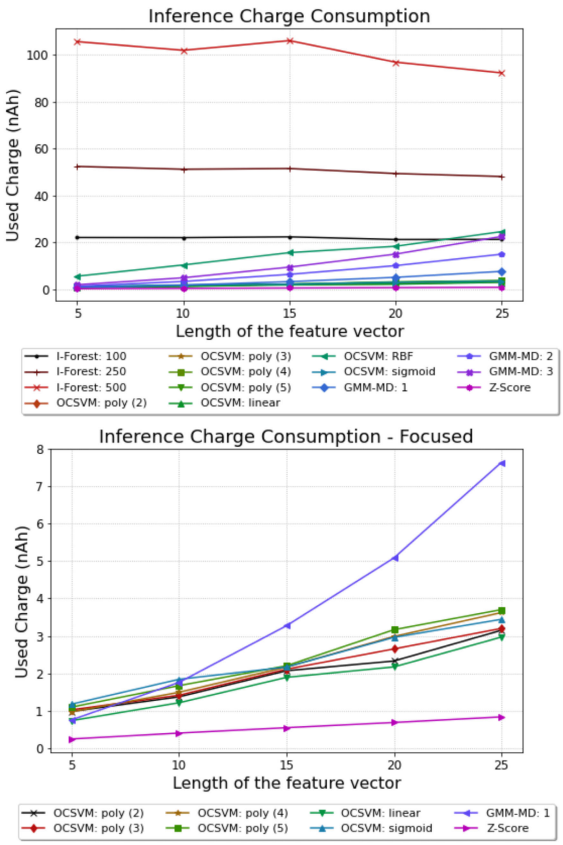


FIGURE 11. Inference charge usage of the anomaly detection algorithms. All algorithms are shown on the top, and the bottom image focuses on the algorithms with the lowest charge.

3) MODEL ACCURACY

The accuracy of an anomaly detection algorithm is given by its ability to correctly identify if the structure's state is damaged or undamaged. There are several metrics to quantify the performance of these algorithms. The most commonly used performance metrics in ML are precision, which measures the rate of all the data points where the structure is detected as damaged that have been correctly identified and is given by Equation 10; recall or sensitivity, given by Equation 11 is the rate of data points from the damaged test case that have been correctly classified; F-score, which is a combination of the recall and precision metrics and is given by Equation 12; specificity is the rate of data points from the undamaged test case that have been correctly classified, and is given by Equation 13.

$$Precision = \frac{TP}{TP + FP} \tag{10}$$

$$Recall = \frac{TP}{TP + FN} \tag{11}$$

$$FScore = 2\frac{Precision \cdot Recall}{Precision + Recall} \tag{12}$$

$$Specificity = \frac{TN}{TN + FP} \tag{13}$$

where:

- True positive (*TP*) are the data points from the damage test case that have been correctly identified.
- False positive (*FP*) are the data points from the undamaged test case, that have been incorrectly classified as coming from a damaged structure.
- True negative (*TN*) are the data points from the undamaged test case that have been correctly identified.
- False negative (*FN*) are the data points from the damaged test case that have been incorrectly classified as coming from an undamaged structure.

To limit the number of graphs shown in this section, the results shown in the graphs are the averaged results obtained for the evaluated algorithms across the different benchmarks. The average precision metric is shown in Figure 12. It can be seen that for the severe damage case, damage test case 1, the OCSVM algorithm, except for the case with the RBF kernel, outperforms the rest. It is followed by the GMM-MD and Z-score algorithms. It can be seen that for the OCSVM, GMM-MD and Z-score algorithm, the precision reaches a plateau as the length of the feature vector increases, meaning the further increments in the length of the feature vector do not increase the precision in a significant way. The OCSVM algorithm reaches the plateau with only 10 features, while the GMM-MD and Zscore algorithms require 20 features to reach the plateau. The I-forest algorithm obtained the worst performance, since its performance decreased when more than 15 features are considered due to overfitting.

On the slight damage test case, damage test case 2, the GMM-MD algorithm has the highest precision, followed by the Z-Score algorithm. Unlike the results obtained for the severe damage test case, where the precision stabilized for long feature vectors; in the slight damage test case, the precision continues to increase with the dimension of the feature vector. Therefore, showing that the models can benefit from adding even more features. This contrasts the results obtained for the OCSVM and I-Forest algorithms, which stabilize their performance at 15 features.

A noticeable effect is that increasing the number of clusters in the GMM-MD algorithm has a negative impact on its performance, since it may be suffering from overfitting. In cases where the structure may be subjected to different stresses or loaded in different ways, adding more clusters can help to discriminate between these cases. Tools such as the Bayesian Information Criterion [61] are commonly used to determine the optimal number of clusters for a given application.

Similar results have been obtained for the average recall metric, Figure 13. It can be seen that for damage test case 2, the recall is mostly below 50% showing that the algorithms have a hard time identifying slight damage cases. This is to be expected as the slight damage cases produce small changes in the response of the structure, thus making them hard to classify. Consequently, the anomaly score value of a single monitoring event is not a good indicator when the structure is slightly damaged.

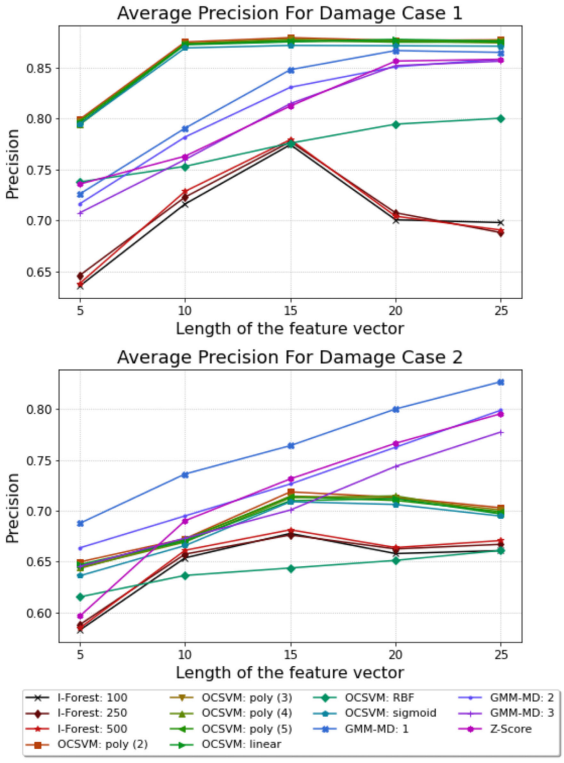


FIGURE 12. Average precision, damage test case 1 on the top, damage test case 2 on the bottom.

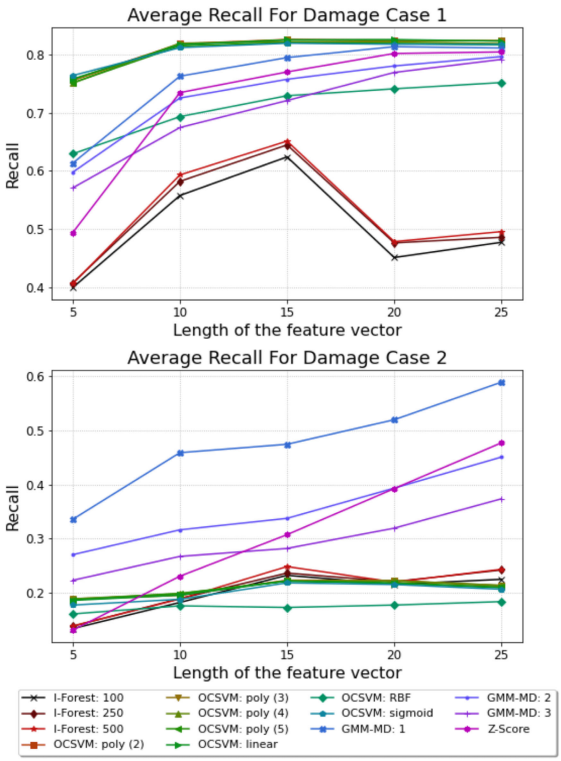


FIGURE 13. Average recall, damage test case 1 on the top, damage test case 2 on the bottom.

The average F-score metric, shown in Figure 14, confirms the initial observations that the OCSVM algorithm

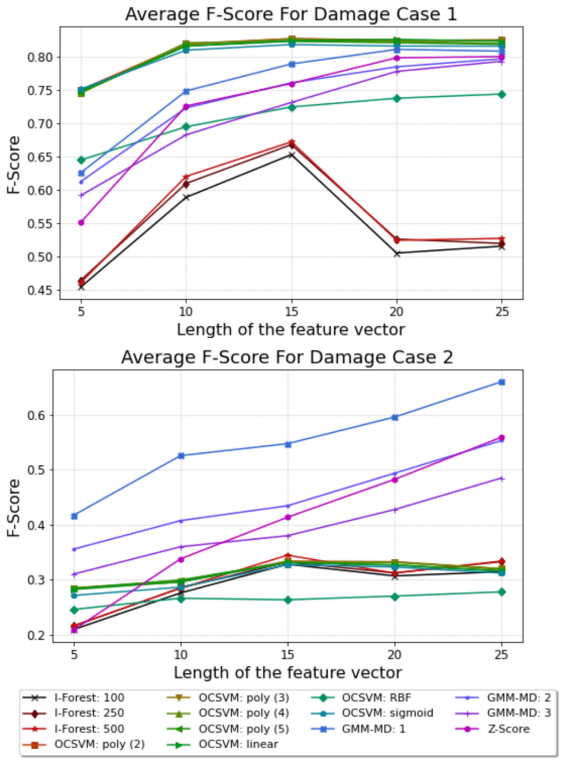


FIGURE 14. Average F-score, damage test case 1 on the top, damage test case 2 on the bottom.

outperforms the rest in the severe damage case; while the GMM-MD algorithm takes the lead for the slight damage test case.

The specificity value only considers how the data points from the undamaged test case are classified. The specificity value is controlled by the percentile parameter used to adjust the anomaly score threshold, which was fixed to 90% for the evaluation. Since the threshold is set using training data, this metric identifies how well the algorithms are able to hold the specificity value for previously unseen data points. Figure 15, shows the average specificity for the evaluated models. It can be seen that in all cases, the value is only slightly lower than 90% regardless of the number of features.

On top of the performance metrics, we also calculated their variability, since each configuration was tested 500 times. As an example of this variability, the standard deviation of the F-score for the damage test case 2 is shown in Figure16. The I-Forest algorithm and the GMM-MD algorithm with multiple clusters had the highest standard deviation. The next algorithm in terms of standard deviation was the GMM-MD with only one cluster, followed by the Z-score algorithm. The OCSVM algorithm had the lowest variability. Taking into account the variance, the F-Score value obtained with the GMM-MD algorithm on the damage test case 2 with 25 features, is higher than 0.461 in over 98% of the cases.

Despite its low memory and processing requirements, the Z-score algorithm has shown promising results. Making it

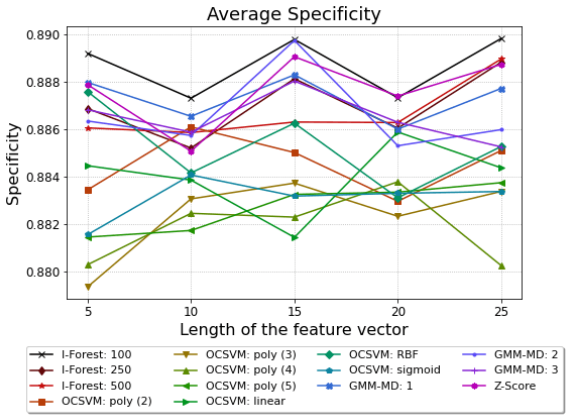


FIGURE 15. Average model specificity.

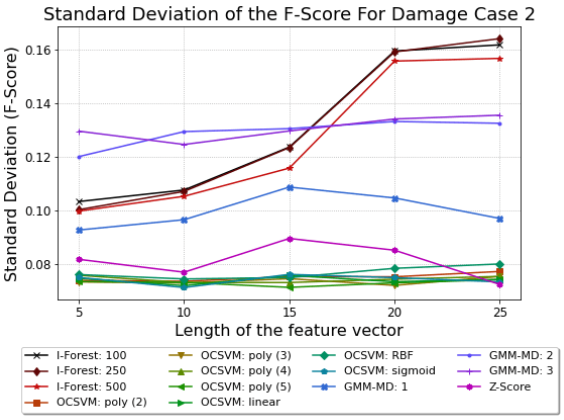


FIGURE 16. Average model F-score standard deviation.

an interesting alternative in cases where the sensor nodes do not have enough resources for the GMM-MD algorithm. From another point of view, given the memory and energy constraints for a sensor node, using the Z-score allows us to use more features, compared to the other algorithms. We evaluated this case by extending the simulations for the Z-score algorithm increasing the number of features. The average F-score for the Z-score algorithm with the extended feature is shown in Figure 17. In this case, only the results for Damage test case 2 are shown, as Figure 14 already showed that the F-score for the severe damage test case reached a plateau at 20 features. The extended simulation shows that increasing the number of features above 25 was actually detrimental to the Z-score algorithm, as the performance peak is reached with 25 features.

The evaluation results per benchmark are shown in Table 7. To limit the size of the table, it only shows the results obtained with 25 features, and from the best-performing hyperparameter for each evaluated algorithm. The results obtained with benchmark 2 (i.e. the SMC bridge benchmark) were significantly worse than for the rest of the benchmarks. The SMC structure is a large and complex structure, which was

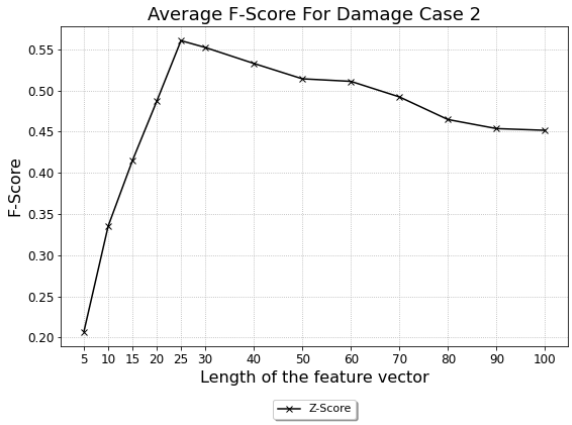


FIGURE 17. Extended feature evaluation for the Z-score algorithm.

subjected to multiple different operating conditions (e.g. wind and traffic) and environmental conditions (e.g. temperature, rain, and humidity) while the benchmark data was recorded. On top of this, the damage conditions were less severe than the cases from the other benchmarks. Consequently, for this benchmark, the anomaly score of a single monitoring event is not a good indicator of the presence of damage in the structure. In these cases, where the changes in the dynamic response of the structure are small, the running percentile of the anomaly scores is a more robust indicator to detect damage.

To test the ability of the anomaly monitoring method of identifying slight damages using the moving percentile of the anomaly scores, we calculated the average relative increment of the 90th percentile of the anomaly scores obtained for the damage test cases with respect to the 90th percentile obtained for the anomaly scores of the undamaged test case. Figure 18, shows the achieved increment in the 90th percentile obtained by the evaluated anomaly detection algorithms. As can be seen, the GMM-MD algorithm outperforms the other algorithms in the severe and the slight damage test cases. For the OCSVM, the polynomial kernel outperforms the other kernels. Unlike the other metrics, such as the F-score value, the polynomial kernels with higher orders obtained a larger increment. For the GMM-MD algorithm with one cluster and 25 features, the relative increments for benchmark 2 is 51.076% and 26.15% for damage test cases 1 and 2, respectively; and their standard deviation of 11.832% and 8.453%. The standard deviation in the undamaged test case was 7.707%. In other words, the increment in the moving 90th percentile of the anomaly scores for the slight damage test case with benchmark 2 is higher than three standard deviations with respect to the 90th percentile value of the undamaged test case. This shows that the moving percentile metric is a good indicator to identify when the structure is damaged even when the change in the structure's response is weak.

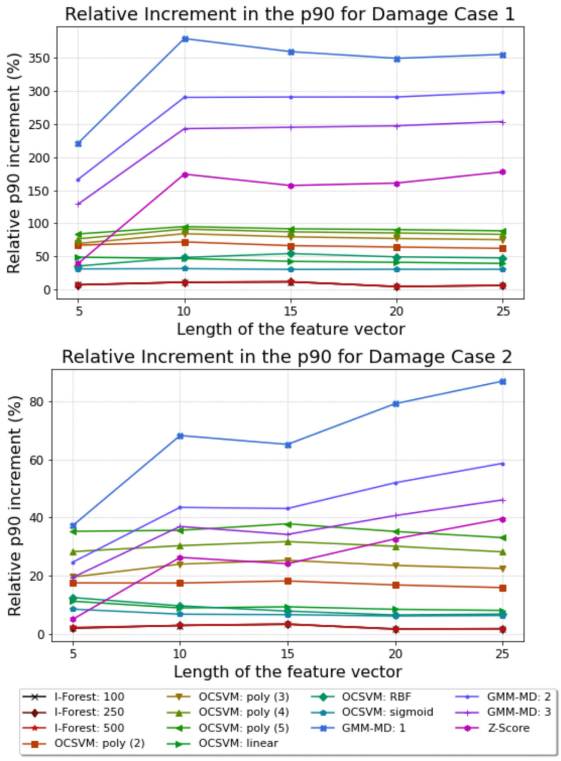


FIGURE 18. Relative increment of the 90th percentile of the anomaly score with respect to the undamaged test case.

4) CONCLUSION OF THE MODEL EVALUATION

Civil structures slowly degrade over time. The goal of an SHM system is to detect the presence of damages as soon as possible, i.e. when the dynamic response of the structure is only slightly altered. This scenario matches the conditions of the damage test case 2, for which the GMM-MD algorithm yielded the best performance, obtaining an average F-score of 0.66 with 25 features. The main drawback of the GMM-MD algorithm is its limited scalability with the number of features. The complexity of the inference is $O\left(mn^2\right)$, and the number of elements in its model $m(n^2+n)$ for m clusters and n features. For the severe damage test case, the algorithms achieved a plateau, beyond which adding additional features did not improve the performance of the algorithms, and thus, the models to identify severe damage can use shorter feature vectors, without sacrificing performance. However, this is not the case for the slight damage test case, as the performance of the GMM-MD algorithm continued to improve as the length of the feature vector is increased. To identify slight damages having a large feature space is beneficial, which significantly impacts the energy and memory efficiency of the GMM-MD algorithm. On the other side, our evaluation results show that using a single cluster gave the best results, which mitigates the energy and memory cost of the GMM-MD algorithm, particularly when the feature vector is large.

The battery charge required to perform a single inference with the GMM-MD algorithm with 25 features is

TABLE 7. Results of the anomaly detection algorithms with the best-performing hyperparameters by benchmark.

Algorithm	Benchmark	Damage test case 1			Damage test case 2				Specificity
		Precision	Recall	F-Score	Precision	Recall	F-Score	σ(FScore)	
I-Forest: 500	benchmark 1	0.43	0.14	0.193	0.605	0.127	0.2	0.126	0.883
	benchmark 2	0.553	0.154	0.23	0.545	0.138	0.213	0.112	0.896
	benchmark 3	0.876	0.689	0.738	0.754	0.33	0.423	0.27	0.885
	benchmark 4	0.904	0.999	0.948	0.778	0.378	0.498	0.118	0.892
OCSVM: poly (2)	benchmark 1	0.936	0.987	0.96	0.823	0.299	0.434	0.051	0.883
	benchmark 2	0.729	0.309	0.423	0.626	0.197	0.291	0.106	0.886
	benchmark 3	0.941	1	0.969	0.775	0.2	0.31	0.078	0.882
	benchmark 4	0.902	0.997	0.946	0.588	0.156	0.241	0.074	0.889
GMM-MD: 1	benchmark 1	0.937	1	0.967	0.911	0.703	0.781	0.121	0.883
	benchmark 2	0.677	0.249	0.352	0.612	0.174	0.264	0.103	0.893
	benchmark 3	0.943	1	0.97	0.936	0.873	0.898	0.064	0.886
	benchmark 4	0.902	0.995	0.945	0.848	0.607	0.699	0.099	0.889
Z-Score	benchmark 1	0.936	1	0.967	0.794	0.253	0.377	0.076	0.882
	benchmark 2	0.647	0.217	0.315	0.597	0.164	0.252	0.102	0.894
	benchmark 3	0.942	1	0.97	0.938	0.896	0.913	0.043	0.884
	benchmark 4	0.906	1	0.95	0.853	0.595	0.696	0.069	0.895

approximately $8nAh$, making this algorithm energy-efficient, at least when compared with the energy used for the collection of the accelerometer data window, which is more than two orders of magnitude higher, in the range of $3\mu Ah$.

Overall, the GMM-MD algorithm showed the best performance, while still being lightweight enough to be implemented in a wireless sensor node, followed by the OCSVM algorithm with a second-order polynomial kernel if the structure is expected to be severely damaged. The results also show that the Z-score algorithm also provides a good tradeoff between energy and memory usage, and the achieved performance, thus making it an interesting alternative for heavily constrained sensor nodes.

V. EXPERIMENTAL EVALUATION IN THE UPC BENCHMARK STRUCTURE

The benchmark evaluation tests used the data from public benchmarks to test different aspects of the anomaly-aware monitoring method presented in this paper. Even though during the evaluations the feature extraction and inference were performed by a sensor node, the data collection and preprocessing were done externally. In this section, we present the experimental evaluation carried out to validate the results from the benchmark evaluations and assess the anomaly-aware monitoring method in a benchmark structure. For the experimental evaluation, the complete anomaly-aware monitoring method from Figure 1, except for the last step, is implemented in the sensor node, which was deployed in a benchmark structure and tested under several damaged and undamaged conditions.

The benchmark structure is a steel frame, reduced-scale model structure, developed and maintained by the Department of Civil and Environmental Engineering at the Polytechnic University of Catalonia (UPC) [62], [63]. The structure has a four-story, two-bay by one-bay configuration, with a height of 2m and a base of 1.45 $\times$ 0.77m, as shown in Figure 19. The horizontal beams of the structure have 4.5kg lead blocks located at the center point of the beam, representing the permanent loads of the structure. This benchmark structure has been used in prior publications to validate structural damage detection methods. Caselles et al. [63] tested a PCA (Principal Component Analysis)-based method, and validated their proposed damage detection method using the same benchmark structure. For their experimental evaluation, Caselles et al. simulated different damage scenarios by releasing the bolts of different junctions.

In our experimental setup, we positioned two sensor nodes attached to the top beam of the front, and back facades of the structure, as close to the center as possible, but slightly to the left as the lead blocks were on the way. The sensor nodes are the same modified commercial nodes by Wordsensing used for the benchmark evaluation tests. Each node weighs approximately 300g and was placed on opposite sides of the structure to avoid unbalancing of the structure. In Figure 19 the position of the sensor nodes is highlighted with a red circle, and the orientation of the accelerometer axes is shown in cyan. The sensor nodes were configured to sample acceleration data at a rate of $1000Hz$, and to collect a 2000 sample window for each monitoring event. The internal accelerometer has a band-pass filter with $-3dB$ corner frequencies of 2.38 mHz and $250Hz$ respectively. Even though the accelerometer data is collected along the three axes, the anomaly-aware monitoring method is only implemented for the Y-axis data.

The monitoring events are generated by striking the base of the structure with a 2.5kg medicine ball, which is swung to the structure from a constant height, so as to induce similar vibration levels to the structure in each monitoring event. The medicine ball striking the structure can be seen in Figure 20 (a), and the strike location is highlighted with a green 16-point star in Figure19. This excitation method is

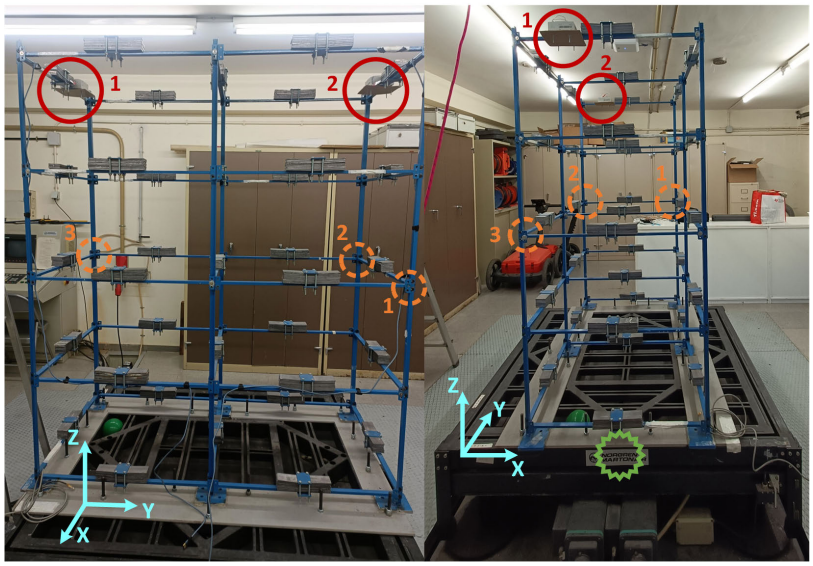


FIGURE 19. UPC benchmark structure. Side view on the left, front view on the right.

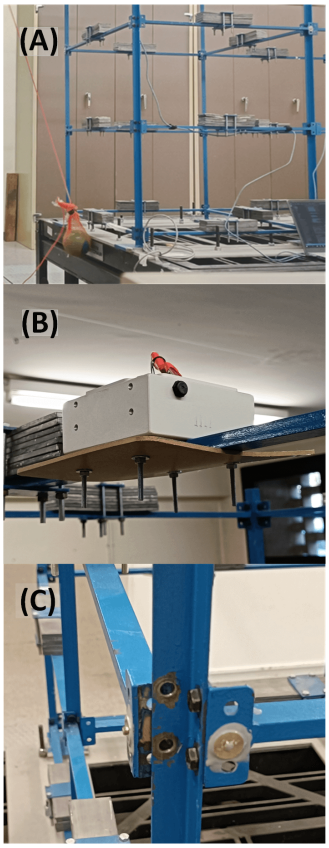


FIGURE 20. The top image (a) shows the medicine ball striking the structure, the center image (b) shows the node attached to the structure, and the bottom image (c) shows the simulated damage on a junction.

the same as the one used by Caselles et al. for their evaluation, showing it does generate enough energy to measure the dynamic response of the structure. When the ball strikes the structure, the sensor nodes are wakened up by their internal accelerometer to start a monitoring event. To eliminate the transitory effects caused by the strike, the sensor nodes were configured to discard the first 500 samples (i.e. waiting half a second) after the strike before starting the data collection process for the monitoring event. Additionally, we waited several seconds between each consecutive strike to ensure that the vibration had stopped before starting a new monitoring event.

TABLE 8. Test scenarios of the experimental evaluation.

Case	Description
0	Undamaged structure.
1	The Y-axis beam on junction 1 is released.
2	Case 1 + Y-axis beam on junction 2 is released.
3	Case 2 + Y-axis beam on junction 3 is released.

The anomaly-aware monitoring method was tested under three different damage scenarios, with incremental damage to the structure. The damage scenarios consisted in removing the junction bolts between the horizontal beams along the Y-axis and the column at different locations of the structure. The locations of the junctions where the damages were introduced are highlighted with dashed orange circles in Figure 19, and an image of a *damaged* junction can be seen in Figure 20 (c). The test scenarios evaluated are summarized in Table 8.

The data collected during the experimental setup is made public in [64]. The dataset is composed of 275 different monitoring events from each sensor node. For each monitoring event, we recorded the raw 3d-accelerometer data in m/s^2, the feature vector calculated by the node, and the anomaly score and prediction value calculated with the anomaly detection model. The first 75 monitoring events, the sensor nodes remained in *training mode*, and thus, for these events, the

TABLE 9. Results from the experimental evaluation.

Node	Test case	Precision	Recall	F-Score	P90 Relative Increment	Specificity
Node 1	Test 1	0.889	0.96	0.923	88.38%	0.88
	Test 2	0.875	0.84	0.857	66.35%	0.88
	Test 3	0.889	0.96	0.923	129.60%	0.88
Node 2	Test 1	0.922	0.94	0.931	78.65%	0.92
	Test 2	0.922	0.94	0.931	111.81%	0.92
	Test 3	0.925	0.98	0.951	144.53%	0.92

feature vector contains the complete feature set, and no inference data was recorded (as the model was yet to be trained).

Once the first 75 monitoring events were recorded, we used this data to perform the feature selection and train the anomaly detection model for each node independently. Following the conclusions from the benchmark evaluation tests, the number of features was set to 20, which was the maximum that could be fit in memory, and we used the GMM-MD algorithm with one cluster to train the model. 75% of the training data was used to train the model, and the other 25% was used to set the anomaly detection threshold. To compare with the results from the benchmark evaluation tests, we fixed the threshold to the 90th percentile of the anomaly scores calculated for the threshold training data.

The training was done in a PC and then communicated to the sensor nodes, transitioning them to *inference mode*. In inference mode, the sensor nodes can run the full anomaly-aware monitoring flow obtaining the anomaly scores and deciding if the data is considered anomalous or not. The results obtained for each test case on each of the nodes are detailed in Table 9. The anomaly scores of the different monitoring events are shown in Figure 21. Each marker represents the anomaly score of a single monitoring event, and the horizontal dashed, red line shows the fixed anomaly detection threshold. It should be noted that the anomaly scores for the training data were obtained later in a PC using the trained model, and not in the sensor node, as the models were not trained at the moment the data was collected. Also, two monitoring events for test case 3 had an anomaly score greater than 10, these values have been cropped out to improve the readability of the image. The results show that both sensor nodes successfully identified the presence of damage in over 75% of the cases, obtaining an F-score of 0.826 and 0.931 depending on the test case. Overall, the Experimental test show that the anomaly-aware monitoring method is lightweight enough to be executed in a commercial sensor node, using its internal MEMS-based accelerometer, and be able to identify when the structure is damaged.

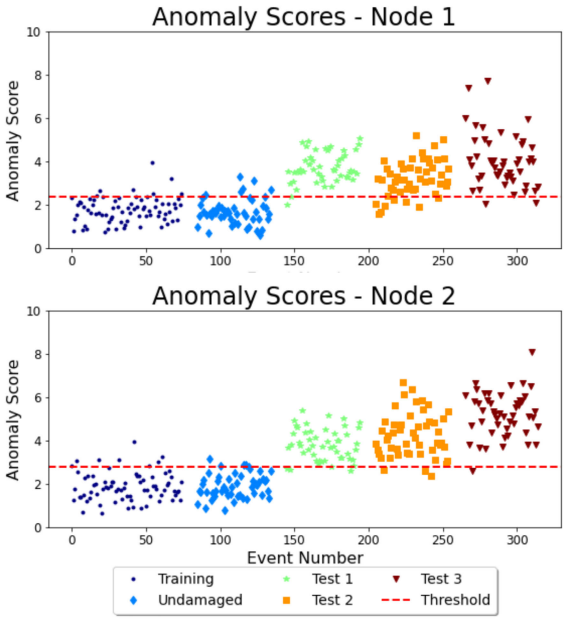


FIGURE 21. Anomaly scores obtained for each monitoring event of the dataset. The results from node 1 are shown on the top, and the results from node 2 are shown on the bottom.

VI. ANOMALY-AWARE MONITORING FOR SELF-AWARENESS

The fourth and last step of the anomaly-aware monitoring workflow presented in Figure 1 is using the anomaly score values to guide the adaptive actions of the sensor node and support the implementation of self-awareness at the sensor node level. The anomaly score value enables the node to react to changes in the monitored structure, without requiring costly and slow communications with the central server. The anomaly score value can be combined with other internal metrics of the node (e.g. its remaining battery life or its energy consumption, etc.) to support the implementation of self-aware behaviors on the sensor node. This section showcases how the anomaly score value can be used to support self-awareness in the context of SHM through an example simulation.

The self-aware monitoring method proposed by Arnaiz et al. [23] dynamically adjusts the power consumption of the sensor node to comply with its defined battery lifetime target; while allocating the energy budget more efficiently by taking into account changes in the temporal correlation of the monitored data. Temporal correlation is a useful parameter to model the perceived utility of the signal for simple measurements, such as temperature, humidity, or wind speed.

However, this metric cannot be used for more complex data such as vibration data in SHM. Instead, we use the anomaly score value to model the perceived utility of the data. Rather than having to transmit all the measurements to the central server, the sensor node can process the data locally, using the anomaly detection model, and only transmit a predefined percentage of values using its radio interface. This way the node can save energy, as it does not have to transmit every monitoring event to the central server, and still be able to detect the presence of local damages in the monitoring structure by analyzing its local anomaly score values.

In the simulation, the node implements the complete anomaly-aware monitoring workflow proposed in this paper. The node remains most of the time in low power mode with just the accelerometer active waiting for vibration peaks to start a monitoring event. These vibration peaks are generated by external factors (e.g. traffic, hammer strikes, trains, etc.) and the node has no control over how often these peaks are generated. But can adjust how often it relays these data to the central server. Once a vibration peak is detected, the node starts a monitoring event, where it collects a window of raw accelerometer measurements, builds the feature vector, and calculates the anomaly score value for the event. The anomaly score value is used to determine if the event data is transmitted, using additional energy, or it is discarded. The self-awareness module needs to dynamically adjust the anomaly score threshold, to control the energy consumption of the sensor node and comply with the battery life target.

To simulate the measurements from the node we used the data from the SMC bridge, or benchmark 2 [39] (Section IV-A2) as it is the more realistic test case from all the benchmarks and contains the time series data from an extended period of time. The benchmark data were preprocessed as described in Section IV; shuffled, so the training period contains data taken at different times with different environmental conditions; and re-timed, introducing random intervals between consecutive samples to simulate the random nature of the monitoring events. To evaluate the response of the sensor node to the presence of damage, we added two groups of 1000 accelerometer windows using data from the periods where the structure is considered to be damaged, from damage test case 2 and damage test case 1, respectively, as defined in Section IV-A2.

The charge consumption of the sensor node is given by Equation 14, where C_{event} is the battery charge consumption of the node since the last monitoring event, I_{idle} is the current consumption of the sensor node in idle mode in mAh, ΔT is the time in hours since the last monitoring event, C_{sp} is the charge required the sample and process the data; and C_{tr} is the charge required to transmit the data, which is set to zero if the measurement is not transmitted in the end.

$$C_{event} = I_{idle}\Delta T + C_{sp} + C_{tr} \tag{14}$$

As the monitoring events are generated using the acceleration trigger and not a fixed monitoring period, the time between events is not known. Consequently, the sensor node cannot directly model its power consumption and predict its remaining battery life. For simplicity, in the simulation, the sensor node models the average time between events, to estimate the expected number of events until the battery lifetime target is reached. The average time between events is calculated using the Exponential Moving Average (EMA) from Equation 15, where α is the smoothing coefficient of the EMA, $\widehat{\Delta T}_{event_x}$ is the mean time between events at time x, and ΔT is the time since the last monitoring event took place.

$$\widehat{\Delta T}_{event_i} = \alpha\Delta T + (1-\alpha)\,\widehat{\Delta T}_{event_{i-1}} \tag{15}$$

Equation 16 is used to calculate the percentage of monitoring events that can be transmitted to the central server while complying with the battery lifetime target. Where C_{bat} is the remaining battery charge, T_{bat} is the time remaining until the battery lifetime target is accomplished, $\widehat{\Delta T}_{event}$ is the mean time between events, and the rest of the parameters are detailed in Table 10. The percentage of monitoring events to be transmitted is used to adjust the anomaly score threshold used to determine if a given monitoring event is transmitted or not. The threshold value is calculated as the moving percentile of the anomaly score values following Algorithm 1.

$$Trans_{\%} = \frac{Cbat - T_{bat}\left(I_{idle} + C_{sp}\widehat{\Delta T}_{event}\right)}{C_{tr}T_{bat}\widehat{\Delta T}_{event}} \tag{16}$$

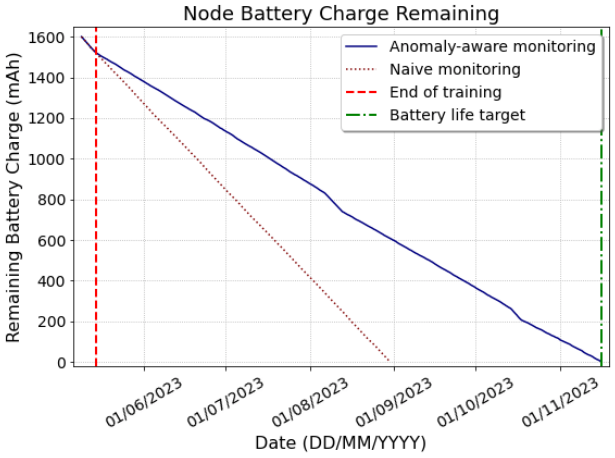


FIGURE 22. Battery discharge rate of a sensor node with the proposed anomaly-aware monitoring method and the naive monitoring method.

TABLE 10. Simulation data for the sensor node.

Parameter	Symbol	Value
Initial Battery charge	C_{bat0}	$1600mAh$
Idle current	I_{idle}	$73.3\mu A$
Sampling + Processing charge	C_{sp}	$3.135\mu Ah$
Radio transmission charge	C_{tr}	$64.689\mu Ah$
Smoothing Coefficient	α	0.005
Target battery life	T_{bat0}	$4585h$

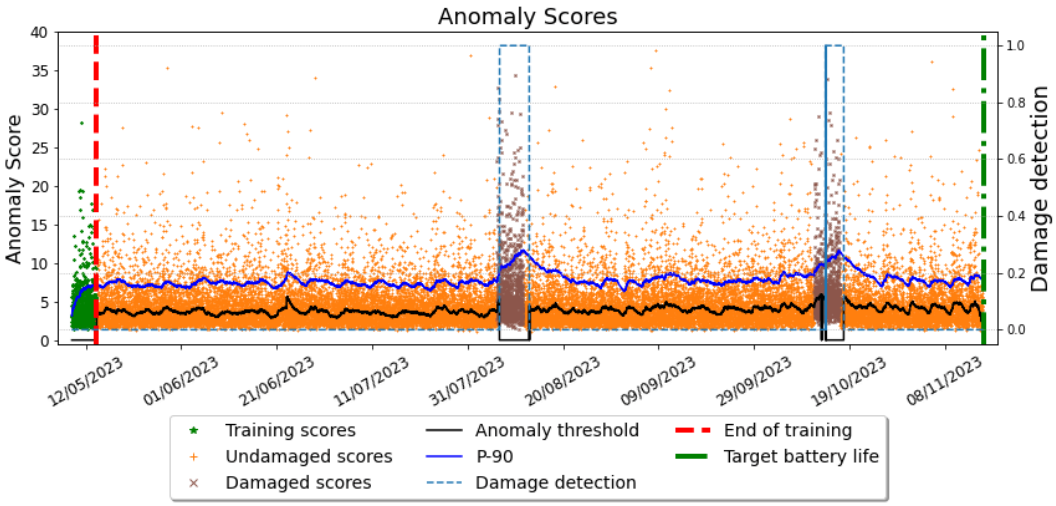


FIGURE 23. Anomaly scores with the anomaly score threshold and the 90th percentile.

Figure 22 shows the sensor node's battery charge over time in the continuous blue line. For the first 1000 monitoring events, the sensor node starts in training mode. The end of the training period is marked with a dashed red vertical line. During the training period, the sensor node transmits the data from all the monitoring events, so they can be used to train the anomaly detection model. For the simulations, we used the GMM-MD model with a single cluster, which is configured in the node once the training period ends. In inference mode, the sensor node computes the anomaly score for each monitoring event to decide if the data is transmitted or not. By not transmitting the information of each monitoring event, the sensor node is able to adjust its energy consumption. As can be seen in Figure 22, after the training is completed, the sensor node reduces its battery discharge rate so it is able to reach the battery lifetime target. For comparison, the discharge rate of a sensor node using a naive monitoring method is shown as a brown dotted line. For the naive monitoring method, the sensor node transmits the data from all the monitoring events, so its energy consumption does not change once the training ends.

The simulation results are detailed in Table 11. The battery life of the node in the simulation is over six months with a mean time between events of eight minutes. The self-awareness module is able to successfully adjust the energy consumption of the sensor node and achieve a battery life within a few hours of its predefined target. Furthermore, by only transmitting the most anomalous monitoring events, the sensor node is able to extend the battery life of the node by 77 days or by 59% compared with the naive monitoring method.

The anomaly scores for the different monitoring events can be seen in Figure 23, along with the 90th percentile of the scores represented as a blue line, and the anomaly score threshold as a continuous black line. All the events with an anomaly score value larger than the anomaly score threshold are transmitted.

TABLE 11. Simulation results. Dates as (DD/MM/YYYY hh:mm:ss).

Result	Value
Start date	09/05/2023 01:53:31
Battery depletion date	16/11/2023 00:31:44
Battery lifetime target	16/11/2023 02:22:41
Naive battery depletion date	30/08/2023 15:51:21
Average time between events	$8min$
Percentage of events transmitted	68%

The monitoring events extracted from when the data is considered to be damaged are shown in brown. As it can be seen when both groups of monitoring events from the damaged structure are being processed, the 90th percentile of the anomaly score increases significantly, allowing the node to identify that the structure is potentially damaged, as it can be seen by the *damage detection* signal, which transitions from zero to one when the node detects the presence of damage. When the node suspects that the structure is potentially damaged, it transmits the information of all the captured monitoring events regardless of its battery life target, as can be seen during this period the anomaly score threshold is set to zero. In the simulation, once the damage period ends the node resumes its regular operation, having to re-adjust its operation to compensate for energy excess consumed during the period where the structure was considered to be damaged.

Overall, the example self-aware application is able to dynamically adjust the percentage of transmitted monitoring events, so as to comply with its target battery life. The anomaly score is leveraged to model which monitoring events are considered more interesting and prioritize the transmission of these events. Moreover, anomaly scores are also used

to detect the presence of structural damage without having to communicate with the central server. Thus, allowing the sensor node to operate with a low percentage of transmitted monitoring events, without impacting the damage detection delay in a significant way.

VII. CONCLUSIONS AND FUTURE WORK

This article proposes a lightweight SHM monitoring method that employs an anomaly detection model to identify variations in the dynamic response of the monitored structure, which may indicate the presence of damages. This article starts evaluating a wide range of commonly used features for SHM using publicly available SHM benchmarks, showing that the Wavelet Packet Component Energy bands outperformed the other features in all cases. Based on the analytic results an unsupervised feature selection method is proposed. The result shows that when the feature vector is large, with 20 or more features, the score of the proposed method is 70% that of the ideal supervised feature selection method.

The anomaly detection model is a critical aspect of the proposed monitoring method. This article evaluates the four most common lightweight anomaly detection algorithms using the data from the SHM benchmarks. The result shows that the GMM-MD algorithm with only one cluster has the best performance in cases where the structure is slightly damaged, while having a relatively small memory footprint and energy cost. For nodes where the resources are heavily constrained, the Z-score algorithm offered a good algorithm, reducing the memory and energy cost by more than 85%, and only reducing the F-score by 20%. The benchmark evaluation also showed that the anomaly score obtained for an individual monitoring event provides a good indication of the presence of severe damage, but is not robust when the structure is slightly damaged. However, using the running percentile of the anomaly score value in consecutive monitoring events, does provide a robust indication even when the damage has very little effect on the dynamic response of the structure.

The evaluation results successfully validated the proposed anomaly-aware monitoring method using a commercial node deployed in a benchmark structure. The results obtained for the evaluation test, running the complete anomaly-aware monitoring method in the sensor nodes, were similar to the results obtained with the public benchmark data, validating that running the algorithm in the node is viable. Furthermore, this article also showcases how the anomaly scores can be used to support self-awareness in a test simulation where the anomaly score values were successfully leveraged to guide the behavior of the sensor node, allowing the sensor node to comply with a specified battery lifetime goal, without impacting the system's ability to identify damages in the structure.

Future research directions will focus on two directions. One research direction is to evaluate the performance of the anomaly detection algorithm in a real structure, under varying environmental conditions. Additionally, the use of environmental measurements (e.g. temperature and humidity) as features also needs to be considered, evaluating their effect on the robustness of the anomaly detection model. The other research direction centers on the implementation of more sophisticated self-aware behaviors at the sensor node level. The self-aware behavior presented in the article is a first approximation, only taking into account one goal (the battery life), two monitored parameters (i.e. remaining battery and anomaly scores), and one control variable (i.e. the transmission rate) to guide the node's behavior. Future self-aware methods should evaluate the use of multiple, competing, goals and multiple control and monitored parameters.

ACKNOWLEDGMENT

The authors wish to thank the Department of Civil and Environmental Engineering at the Polytechnic University of Catalonia (UPC), and in particular Oriol Caselles for granting us access to the UPC benchmark structure to perform the experimental evaluation.

REFERENCES

[1] *Structurally Deficient Bridges | Bridge Infrastructure | ASCE's 2021 Infrastructure Report Card*, ASCE, Reston, VA, UA, 2021.
[2] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Comput. Netw.*, vol. 52, no. 12, pp. 2292–2330, Aug. 2008.
[3] I. F. Akyildiz and M. C. Vuran, *Wireless Sensor Networks*. Hoboken, NJ, USA: Wiley, 2010.
[4] M. Abdulkarem, K. Samsudin, F. Z. Rokhani, and M. F. A. Rasid, "Wireless sensor network for structural health monitoring: A contemporary review of technologies, challenges, and future direction," *Struct. Health Monitor.*, vol. 19, no. 3, pp. 693–735, May 2020.
[5] A. Sofi, J. Jane Regita, B. Rane, and H. H. Lau, "Structural health monitoring using wireless smart sensor network—An overview," *Mech. Syst. Signal Process.*, vol. 163, Jan. 2022, Art. no. 108113.
[6] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon, "Health monitoring of civil infrastructures using wireless sensor networks," in *Proc. 6th Int. Symp. Inf. Process. Sensor Netw.*, Apr. 2007, pp. 254–263.
[7] M. Abner, P. K.-Y. Wong, and J. C. P. Cheng, "Battery lifespan enhancement strategies for edge computing-enabled wireless Bluetooth mesh sensor network for structural health monitoring," *Autom. Construct.*, vol. 140, Aug. 2022, Art. no. 104355.
[8] A. B. Noel, A. Abdaoui, T. Elfouly, M. H. Ahmed, A. Badawy, and M. S. Shehata, "Structural health monitoring using wireless sensor networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1403–1423, 3rd Quart., 2017.
[9] G. Anastasi, M. Conti, M. Di Francesco, and A. Passarella, "Energy conservation in wireless sensor networks: A survey," *Ad Hoc Netw.*, vol. 7, no. 3, pp. 537–568, May 2009.
[10] J. Singh, R. Kaur, and D. Singh, "A survey and taxonomy on energy management schemes in wireless sensor networks," *J. Syst. Archit.*, vol. 111, Dec. 2020, Art. no. 101782.
[11] P. R. Lewis, M. Platzner, B. Rinner, J. Trresen, and X. Yao, *Self-Aware Computing Systems: An Engineering Approach*, 1st ed. Springer, Aug. 2016, p. 354.
[12] J. Zhou and D. De Roure, "FloodNet: Coupling adaptive sampling with energy aware routing in a flood warning system," *J. Comput. Sci. Technol.*, vol. 22, no. 1, pp. 121–130, Jan. 2007.
[13] A. Agarwal, J. Miller, J. Eastep, D. Wentziaff, and H. Kasture, "Self-aware computing," Massachusetts Inst. Technol., Cambridge, MA, USA, Tech. Rep. ADA501367, 2009.
[14] S. Kounev et al., "The notion of self-aware computing," in *Self-Aware Computing Systems*. Cham, Switzerland: Springer, 2017, pp. 3–16, doi: 10.1007/978-3-319-47474-8_1.
[15] M. Götzinger, D. Juhász, N. Taherinejad, E. Willegger, B. Tutzer, P. Liljeberg, A. Jantsch, and A. M. Rahmani, "RoSA: A framework for modeling self-awareness in cyber-physical systems," *IEEE Access*, vol. 8, pp. 141373–141394, 2020.

[16] K. Bellman, C. Landauer, N. Dutt, L. Esterle, A. Herkersdorf, A. Jantsch, N. TaheriNejad, P. R. Lewis, M. Platzner, and K. Tammemäe, "Self-aware cyber-physical systems," *ACM Trans. Cyber-Physical Syst.*, vol. 4, no. 4, pp. 1–26, Oct. 2020.

[17] A. Jantsch, N. Dutt, and A. M. Rahmani, "Self-awareness in systems on chip—A survey," *IEEE Des. Test.*, vol. 34, no. 6, pp. 8–26, Dec. 2017.

[18] E. Gelenbe, J. Domanska, P. Fröhlich, M. P. Nowak, and S. Nowak, "Self-aware networks that optimize security, QoS, and energy," *Proc. IEEE*, vol. 108, no. 7, pp. 1150–1167, Jul. 2020.

[19] Y. Zhuang, L. Yu, H. Shen, W. Kolodzey, N. Iri, G. Caulfield, and S. He, "Data collection with accuracy-aware congestion control in sensor networks," *IEEE Trans. Mobile Comput.*, vol. 18, no. 5, pp. 1068–1082, May 2019.

[20] F. Forooghifar, A. Aminifar, and D. Atienza, "Resource-aware distributed epilepsy monitoring using self-awareness from edge to cloud," *IEEE Trans. Biomed. Circuits Syst.*, vol. 13, no. 6, pp. 1338–1350, Dec. 2019.

[21] J. S. Preden, K. Tammemäe, A. Jantsch, M. Leier, A. Riid, and E. Calis, "The benefits of self-awareness and attention in fog and mist computing," *Computer*, vol. 48, no. 7, pp. 37–45, Jul. 2015.

[22] A. Anzanpour, I. Azimi, M. Götzinger, A. M. Rahmani, N. TaheriNejad, P. Liljeberg, A. Jantsch, and N. Dutt, "Self-awareness in remote health monitoring systems using wearable electronics," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2017, pp. 1056–1061.

[23] D. Arnaiz, F. Moll, E. Alarcón, and X. Vilajosana, "Data relevance-aware dynamic sensing technique with battery lifetime guarantee for wireless sensor nodes," in *Proc. 36th Conf. Design Circuits Integr. Syst. (DCIS)*, Nov. 2021, pp. 1–6.

[24] C. R. Farrar and K. Worden, "An introduction to structural health monitoring," *Philos. Trans. Roy. Soc. A, Math., Phys. Eng. Sci.*, vol. 365, pp. 303–315, Dec. 2006.

[25] A. Malekloo, E. Ozer, M. AlHamaydeh, and M. Girolami, "Machine learning and structural health monitoring overview with emerging technology and high-dimensional data source highlights," *Struct. Health Monitor.*, vol. 21, no. 4, pp. 1906–1955, Jul. 2022.

[26] E. P. Carden and J. M. W. Brownjohn, "ARMA modelled time-series classification for structural health monitoring of civil infrastructure," *Mech. Syst. Signal Process.*, vol. 22, no. 2, pp. 295–314, Feb. 2008.

[27] H. Sohn and K. H. Law, "A Bayesian probabilistic approach for structure damage detection," *Earthq. Eng. Struct. Dyn.*, vol. 26, no. 12, pp. 1259–1281, 1997.

[28] J.-G. Han, W.-X. Ren, and Z.-S. Sun, "Wavelet packet based damage identification of beam structures," *Int. J. Solids Struct.*, vol. 42, no. 26, pp. 6610–6627, Dec. 2005.

[29] H. Sarmadi and A. Karamodin, "A novel anomaly detection method based on adaptive mahalanobis-squared distance and one-class kNN rule for structural health monitoring under environmental effects," *Mech. Syst. Signal Process.*, vol. 140, Jun. 2020, Art. no. 106495.

[30] P. P. Ray, "A review on TinyML: State-of-the-art and prospects," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 34, no. 4, pp. 1595–1623, Apr. 2022.

[31] R. Domingues, M. Filippone, P. Michiardi, and J. Zouaoui, "A comparative evaluation of outlier detection algorithms: Experiments and analyses," *Pattern Recognit.*, vol. 74, pp. 406–421, Feb. 2018.

[32] C. McKinnon, J. Carroll, A. McDonald, S. Koukoura, D. Infield, and C. Soraghan, "Comparison of new anomaly detection technique for wind turbine condition monitoring using gearbox SCADA data," *Energies*, vol. 13, no. 19, p. 5152, Oct. 2020.

[33] Y. Yang, Y. Zhang, and X. Tan, "Review on vibration-based structural health monitoring techniques and technical codes," *Symmetry*, vol. 13, no. 11, p. 1998, Oct. 2021.

[34] A. Sabato, C. Niezrecki, and G. Fortino, "Wireless MEMS-based accelerometer sensor boards for structural vibration monitoring: A review," *IEEE Sensors J.*, vol. 17, no. 2, pp. 226–235, Jan. 2017.

[35] LoRa Alliance. *What is LoRaWAN Specification*. Accessed: Jul. 20, 2023. [Online]. Available: https://lora-alliance.org/about-lorawan/

[36] O. Abdeljaber, A. Younis, O. Avci, N. Catbas, M. Gul, O. Celik, and H. Zhang, "Dynamic testing of a laboratory stadium structure," in *Proc. Geotechnical Structural Eng. Congr.*, Feb. 2016, pp. 1719–1728.

[37] O. Abdeljaber, O. Avci, S. Kiranyaz, M. Gabbouj, and D. J. Inman, "Real-time vibration-based structural damage detection using one-dimensional convolutional neural networks," *J. Sound Vibrat.*, vol. 388, pp. 154–170, Feb. 2017.

[38] O. Avci, O. Abdeljaber, S. Kiranyaz, M. Hussein, M. Gabbouj, and D. Inman, "A new benchmark problem for structural damage detection: Bolt loosening tests on a large-scale laboratory structure," in *Dynamics of Civil Structures, Volume 2*. Berlin, Germany: Springer, 2022, pp. 15–22.

[39] S. Li, H. Li, Y. Liu, C. Lan, W. Zhou, and J. Ou, "SMC structural health monitoring benchmark problem using monitored data from an actual cable-stayed bridge," *Struct. Control Health Monitor.*, vol. 21, no. 2, pp. 156–172, Feb. 2014.

[40] S. J. Dyke, D. Bernal, J. Beck, and C. Ventura, "Experimental phase II of the structural health monitoring benchmark problem," in *Proc. 16th ASCE Eng. Mech. Conf.*, 2003, pp. 1–7.

[41] J.-Z. Huang, D.-S. Li, H.-N. Li, G.-B. Song, and Y. Liang, "Damage identification of a large cable-stayed bridge with novel cointegrated Kalman filter method under changing environments," *Struct. Control Health Monitor.*, vol. 25, no. 5, p. e2152, May 2018.

[42] G. De Roeck, B. Peeters, and J. Maeck, "Dynamic monitoring of civil engineering structures," Dept. Civil Eng., K.U.Leuven, Leuven, Belgium, Jul. 2000. [Online]. Available: https://bwk.kuleuven.be/apps/bwm/papers/deroip00b.pdf

[43] B. Peeters and G. De Roeck, "One-year monitoring of the Z24-bridge: Environmental effects versus damage events," *Earthq. Eng. Struct. Dyn.*, vol. 30, no. 2, pp. 149–171, 2001.

[44] A. Teughels and G. De Roeck, "Structural damage identification of the highway bridge Z24 by FE model updating," *J. Sound Vibrat.*, vol. 278, no. 3, pp. 589–610, Dec. 2004.

[45] *K.U. Leuven webpage for the Z24 Bridge Benchmark*. Accessed: Dec. 19, 2022. [Online]. Available: https://bwk.kuleuven.be/bwm/z24

[46] *Documentation Page for the Qatar University Grandstand Simulator (QUGS) Benchmark*. Accessed: Dec. 19, 2022. [Online]. Available: http://onur-avci.com/benchmark/

[47] Q. Gu, Z. Li, and J. Han, "Generalized Fisher score for feature selection," 2012, *arXiv:1202.3725*.

[48] M. A. Cody, "The wavelet packet transform: Extending the wavelet transform," *Dr. Dobb's J.*, vol. 19, pp. 44–46, Apr. 1994.

[49] Z. Sun and C. C. Chang, "Structural damage assessment based on wavelet packet transform," *J. Struct. Eng.*, vol. 128, no. 10, pp. 1354–1361, Oct. 2002.

[50] Y. Ding, A. Li, and T. Liu, "A study on the WPT-based structural damage alarming of the ASCE benchmark experiments," *Adv. Struct. Eng.*, vol. 11, no. 1, pp. 121–127, Feb. 2008.

[51] X.-L. Peng, H. Hao, and Z.-X. Li, "Application of wavelet packet transform in subsea pipeline bedding condition assessment," *Eng. Struct.*, vol. 39, pp. 50–65, Jun. 2012.

[52] R. Hou and Y. Xia, "Review on the new development of vibration-based damage identification for civil engineering structures: 2010–2019," *J. Sound Vibrat.*, vol. 491, Jan. 2021, Art. no. 115741.

[53] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *Proc. 8th IEEE Int. Conf. Data Mining*, Dec. 2008, pp. 413–422.

[54] B. Schölkopf, R. C. Williamson, A. Smola, J. Shawe-Taylor, and J. Platt, "Support vector method for novelty detection," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 12, 1999, pp. 1–7.

[55] D. A. Reynolds, "Gaussian mixture models," *Encyclopedia Biometrics*, vol. 741, nos. 659–663, Jul. 2009.

[56] P. C. Mahalanobis, "On the generalized distance in statistics," in *Proc. Nat. Inst. Sci. India*. Jatni, India: National Institute of Science of India, Apr. 1936.

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

[58] Worldsensing. *Tilt90-x Information Page*. Accessed: Jul. 20, 2023. [Online]. Available: https://www.worldsensing.com/product/tilt90-x-2/

[59] Silicon Labs. *EFM32GG12 Family Datasheet*. Accessed: Jul. 20, 2023. [Online]. Available: https://www.silabs.com/documents/public/data-sheets/efm32gg12-datasheet.pdf

[60] Texas Instruments. *INA219 Datasheet*. Accessed: Jul. 20, 2023. [Online]. Available: https://www.ti.com/lit/gpn/ina219

[61] A. A. Neath and J. E. Cavanaugh, "The Bayesian information criterion: Background, derivation, and applications," *Wiley Interdiscipl. Rev., Comput. Statist.*, vol. 4, no. 2, pp. 199–203, 2012.

[62] O. Caselles, A. Martín, Y. F. Vargas-Alzate, R. Gonzalez-Drigo, and J. Clapés, "Estimators for structural damage detection using principal component analysis," *Heritage*, vol. 5, no. 3, pp. 1805–1818, 2022.

[63] J. O. Caselles, A. Martín, and J. Clapés, "Detection of damage using temporal variation in natural frequencies and principal component analysis," *Int. J. Architectural Heritage*, vol. 16, no. 4, pp. 616–629, Apr. 2022.
[64] D. Arnaiz, E. Alarcón, F. Moll, and X. Vilajosana, "UPC benchmark structure," Tech. Rep., Jun. 2023. [Online]. Available: https://zenodo.org/record/7992523, doi: 10.5281/zenodo.7992523.

DAVID ARNAIZ received the B.Sc. degree in industrial electronic engineering from the University of Granada (UGR), in 2012, and the M.Sc. degree (Hons.) in electronic engineering from Universitat Politècnica de Catalunya (UPC), in 2018. He is currently pursuing the Ph.D. degree. He was a part of the industrial doctorate program working in collaboration between the UPC and WordSensing. His research interests include dynamic energy management and the introduction of self-awareness in wireless sensor nodes. He received the Best Academic Record Award for his Ph.D. degree.

EDUARD ALARCÓN (Member, IEEE) received the Ph.D. degree, in 2000 and the EE engineering degree from Universitat Politècnica de Catalunya (UPC), Barcelona, Spain. He is a Professor with UPC. His research interests include resource-constrained short-range wireless communications, spacecraft-distributed architectures, AI chip coprocessor architectures, quantum computing architectures, and structured design of complex systems. He has served in different positions at IEEE. He received the National Award for his EE engineering degree.

FRANCESC MOLL (Senior Member, IEEE) received the M.Sc. degree in physics from the University of Balearic Islands, Spain, in 1991, and the Ph.D. degree in electronic engineering from Universitat Politècnica de Catalunya (UPC), in 1995. He has been a Professor with the Department of Electronic Engineering, UPC, since 1997. His research interests include reliability and robustness issues relevant to integrated circuit design especially in advanced technology nodes, such as signal integrity modeling and its impact, manufacturing variability, and ultra-low-power and voltage circuits.

XAVIER VILAJOSANA (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees in computer science from the Universitat Politècnica de Catalunya and the Ph.D. degree in computer science from the Universitat Oberta de Catalunya (UOC). He has been a Researcher with Orange Labs, UC Berkeley, and HP Labs. He is currently a Professor with UOC. His research interests include wireless communications and standardization.