

Department of Physics
University of Colombo

Bridge Health Monitoring System Using Arduino Nano 33 BLE Sense

Name: Iwanthi Abeysinghe

Index: S16083

Date: 31.07.2025

1. Abstract

This project presents the development of a real-time bridge health monitoring system using the Arduino Nano 33 BLE Sense and Edge Impulse. The primary objective is to detect structural anomalies by classifying vibrations as either “Normal” or “Damage” to ensure timely maintenance and prevent failures. A machine learning model was trained on vibration data using Edge Impulse and deployed to the Nano 33 BLE Sense, which continuously collects data via its onboard accelerometer and performs on-device classification. The results are transmitted over USB to a Python Flask server that timestamps and exposes the predictions through a web API. A custom web dashboard updates every two seconds, displaying results in a color-coded table to distinguish between normal and damaged states clearly. The system demonstrated reliable real-time classification, making it a cost-effective and scalable solution for continuous structural health monitoring. It also highlights the potential of embedded AI in critical infrastructure safety applications.

2. Introduction

Bridges play a vital role in transportation infrastructure, enabling the smooth and safe movement of people and goods. Over time, these structures are exposed to a range of stresses such as heavy traffic loads, vibrations, weather effects, and material fatigue. If left unmonitored, these factors can contribute to gradual deterioration or sudden structural failure, posing serious safety risks. Traditional bridge inspection methods, while still widely used, are typically manual, labor-intensive, and infrequent. They often fail to detect early-stage damage and are not suited for real-time monitoring, making them less effective in preventing unexpected failures.

To address these limitations, this project introduces a low-cost, real-time structural health monitoring system using the Arduino Nano 33 BLE Sense microcontroller. The system takes advantage of its onboard LSM9DS1 9-axis IMU sensor to collect vibration data and classify the condition of the bridge as either “Normal” or “Damage”. A machine learning model, trained using Edge Impulse, is deployed directly onto the microcontroller, enabling on-device inference without the need for constant cloud connectivity. This embedded AI approach allows for continuous, local health monitoring of the bridge, even in remote or bandwidth-limited environments.

The classified results are transmitted via USB to a Python Flask server, which timestamps and exposes the data through a web-accessible API. A responsive web dashboard, developed with HTML and JavaScript, visualizes the condition of the bridge in real-time, updating every two seconds with intuitive color-coded indicators. This system provides an accessible and efficient solution for monitoring structural integrity, showcasing how embedded machine learning can enhance public safety and infrastructure reliability through real-time, automated condition assessment.

3. Methodology

System Components

- **Microcontroller:** Arduino Nano 33 BLE Sense
- **Sensor:** LSM9DS1 (9-axis IMU: accelerometer, gyroscope, magnetometer)
- **Communication:** USB Serial
- **Software Tools:** Edge Impulse, Arduino IDE, Python Flask, HTML/JavaScript

The development of the bridge health monitoring system followed a structured approach that combined hardware components, embedded machine learning, and web-based visualization. The Arduino Nano 33 BLE Sense, equipped with the LSM9DS1 IMU, served as the core device for collecting motion data. The accelerometer within the IMU, known for its high sensitivity and precision, was used to detect vibration patterns on the surface of a model bridge. The device was securely mounted to the bridge to ensure accurate capture of vibrational signals under different conditions.

To train the system to classify structural health states, data were gathered under two controlled conditions. In the "Normal" state, the bridge remained undisturbed, while in the "Damage" state, physical taps or shocks were introduced to simulate faults. The raw vibration data collected via USB serial communication was imported into Edge Impulse, a machine learning platform designed specifically for embedded systems. Within Edge Impulse Studio, the data underwent preprocessing, including noise reduction and feature extraction using Fast Fourier Transform (FFT). A classification model was then trained to differentiate between normal and damaged states, and its performance was validated using accuracy metrics and a confusion matrix. The final, optimized model was exported as a C++ library compatible with the Arduino platform.

This model was integrated into custom firmware developed using the Arduino IDE. The firmware continuously read accelerometer data and ran the trained model locally on the microcontroller to provide real-time classification. Each prediction result, either "Normal" or "Damage," was sent to a connected computer via USB serial.

On the host computer, a Python Flask server was developed to handle incoming serial data, timestamp each classification result, and make it accessible via a RESTful API. A live dashboard was created using HTML, CSS, and JavaScript to fetch these predictions at two-second intervals and display them in a dynamic, color-coded table. Green indicated a "Normal" state, and red signaled "Damage," allowing users to visually monitor the bridge's condition in real time.

To verify the functionality and reliability of the system, tests were conducted across multiple scenarios. The system consistently provided accurate predictions, and the dashboard updated in real time without delays. The classification logs were also stored for further analysis. This methodology demonstrates how embedded sensing, machine learning, and real-time web technologies can be effectively integrated into a scalable bridge health monitoring solution.

4. Results

The classification results were transmitted to the connected computer and were received correctly by the Python Flask server. The web dashboard displayed these results promptly, updating every two seconds with the latest prediction along with the corresponding timestamp. Visual feedback on the dashboard was clear and effective, with green indicating a “Normal” state and red indicating a “Damage” state, allowing for immediate interpretation of the health status of the bridge.

Table 1 below shows a sample of the prediction output recorded during testing:




 Date	 Time	 Prediction
2025-07-23	16:29:22	Normal
2025-07-23	16:29:27	Damage
2025-07-23	16:29:32	Damage
2025-07-23	16:29:37	Damage
2025-07-23	16:29:42	Normal
2025-07-23	16:29:47	Damage
2025-07-23	16:29:52	Damage

Table 1: Sample Real-Time Classification Output

Throughout testing, the system consistently demonstrated reliable operation and accurate classification. The real-time feedback mechanism and visual alert system proved effective in providing intuitive and continuous monitoring of the structural state of the model.

5. Discussion

This project aimed to develop a real-time bridge health monitoring system using an embedded AI model capable of detecting structural vibrations and classifying the state as either “Normal” or “Damage.” The system successfully demonstrated its ability to process vibration data locally and provide immediate feedback via a web-based dashboard, validating the feasibility of combining edge computing with machine learning for structural health diagnostics.

The results indicated that the embedded model could accurately classify bridge conditions during controlled testing, confirming the effectiveness of using accelerometer data for detecting anomalies. The real-time dashboard functioned reliably, providing intuitive visual alerts that made it easy to distinguish between normal and abnormal states. These findings support the use of low-cost microcontrollers with onboard sensors and AI models for continuous condition monitoring.

However, the performance of the system is closely tied to the quality and diversity of the training data. Since the model was trained using only two controlled conditions on a model bridge, its

ability to generalize to more complex or real-world scenarios may be limited. Factors such as environmental noise, variable load conditions, and different structural materials were not included in the training set, which could affect the robustness of the system in field deployment.

Compared to existing literature on structural health monitoring systems, the results are promising but highlight some limitations. One notable constraint is the reliance on USB serial communication for data transmission, which restricts the portability and remote usability of the system. In practical applications, wireless communication technologies such as Bluetooth or Wi-Fi would be more suitable to enable remote monitoring without tethered connections. Additionally, the use of a single accelerometer limits the spatial coverage and sensitivity. Integrating multiple sensors or expanding the feature set with gyroscopic data could enhance detection accuracy and fault localization.

Despite these limitations, the system meets its primary objective that embedded machine learning can be effectively applied to monitor bridge health in real time. It lays a solid foundation for developing scalable and autonomous diagnostic tools for civil infrastructure.

Future improvements could include training the model with a wider variety of vibration patterns, incorporating environmental filtering algorithms, adding multi-sensor fusion, and transitioning to wireless communication. These enhancements would increase the robustness, adaptability, and readiness of the system for real-world deployment.

6. Conclusion

This project aimed to create a system that can monitor bridge health in real time using embedded AI. The system worked well by accurately detecting normal and damaged states through vibration data. This shows that low-power devices with machine learning can help keep bridges safe. With some improvements, such systems could be used for early damage detection and remote monitoring to protect public safety.

7. Reference

- *Bridge Quality Dataset*. (2024, September 9). Kaggle.
<https://www.kaggle.com/datasets/aayushmansharma13/bridge-quality-dataset>
- *Nano BLE inference with Accelerometer*. (2025, June 27). Edge Impulse Forum.
<https://forum.edgeimpulse.com/t/nano-ble-inference-with-accelerometer/14392>

