International Journal of Innovative
Computing, Information and Control
Volume **22**, Number **2**, April **2026**

ICIC International ©2026 ISSN 1349-4198
pp. 565–578

LIGHTWEIGHT CNN-BASED BEARING FAULT DIAGNOSIS ON RESOURCE-CONSTRAINED DEVICES

Amos Tan[1] and Chatchai Wannaboon[2,*]

[1]School of Engineering
Temasek Polytechnic
Singapore 529757, Singapore
amostan2359@gmail.com

[2]Intelligent Electronics System Laboratory
Thai-Nichi Institute of Technology
Bangkok 10250, Thailand
[*]Corresponding author: chatchai@tni.ac.th

Received August 2025; revised December 2025

Abstract. *Detecting faults early in rotating machine bearings is important to preventing accidents and costly maintenance downtime. Traditional monitoring systems often require bulky data acquisition equipment and cloud-based analysis, which are unsuitable for compact or remote applications. This paper presents a lightweight Convolutional Neural Network (CNN)-based approach for real-time anomaly detection of motor shaft and bearing unbalance, fully implemented on resource-constrained embedded systems. A Raspberry Pi Pico4ML microcontroller, equipped with an onboard inertial measurement unit (IMU), is mounted directly on the motor to capture vibration data during operation. Vibration data collected from the normal and anomalous bearings were standardized, labelled and converted into windowed time-series data. Segmenting the time-series data into fixed-size windows enables the model to capture temporal patterns effectively, thereby facilitating faster and more accurate predictions. The CNN architecture was trained and quantized with a focus on balancing accuracy and compactness for deployment on the Pico4ML. Experimental results demonstrate a high accuracy in distinguishing the normal and anomaly class on an unseen test dataset. The proposed approach offers a low-cost and easily replicable solution for developing an edge-based anomaly detection system for machine bearings.*

Keywords: Anomaly detection, Deep learning, Edge computing, Convolutional Neural Network

1. **Introduction.** Rotating machinery such as motors plays a vital role in a wide range of industrial applications. Over time, mechanical components like shafts and bearings are subject to wear, misalignment, and unbalance, which can lead to performance degradation, increased energy consumption, and unexpected system failures. Early detection of such anomalies is crucial for predictive maintenance and operational efficiency. Conventional fault detection systems typically rely on high-performance computing platforms and cloud-based analytics, which are often impractical for embedded or real-time applications due to their cost, latency and data vulnerability [1, 2]. Generally, anomaly detection in bearings can be categorized into two main approaches, i.e., model-based and data-driven methods [3]. Model-based methods identify anomalies in bearings by calculating and processing the values that are obtained from the machine, which can be extremely complicated with no prior knowledge [4]. On the other hand, data-driven methods involving AI have

DOI: 10.24507/ijicic.22.02.565

widely gained attention for their adaptability and effectiveness in recent years. AI based approaches can be divided into machine learning (ML) and deep learning (DL) [5]. ML methods such as random forest [6, 7], support vector machine (SVM) [8, 9] and k-nearest neighbours (k-NN) [10, 11] learns the characteristics of different data and can identify faults. However, it struggles in learning complex nonlinear relations between data without the use of proper preprocessing techniques. In contrast, DL techniques such as CNN [12, 13], recurrent neural networks (RNN) [14] and Long Short-Term Memory (LSTM) [15] are more effective at learning complex relations hidden in data, as they utilize deep neural networks to extract relevant features [16]. Especially when dealing with raw time-series data such as vibration signals, DL methods can efficiently extract meaningful features and make decisions quickly [17]. Unsupervised learning models such as autoencoders are highly effective for fault detection in bearings, particularly in real-world scenarios where labeled anomalous data are unavailable [18]. Leroux and Simoens introduce a hybrid approach consisting of a supervised and unsupervised model for anomaly prediction [19]. In this system, the unsupervised models were deployed onto edge devices and perform the initial anomaly detection. Upon detecting an anomaly, the data are transmitted to a centralized cloud-based system, where a supervised model is employed to further train and validate the anomaly. This two-stage approach enhances the overall accuracy of anomaly detection by verifying the edge device's initial predictions. However, transferring critical data from edge devices to the cloud introduces potential security risks, particularly if appropriate encryption and data protection mechanisms are not implemented.

Recent advancements in microprocessor technology have significantly enhanced the computational capabilities of edge devices, enabling them to run deep learning (DL) models directly on-device. These improvements allow edge devices to perform real-time data processing and inference without relying on remote servers, thereby supporting low-latency, energy-efficient, and autonomous intelligent systems across a wide range of applications. When applied with proper sensors, EDs enable efficient on-device inference, eliminating the need for continuous data transmission to external servers. Their ability to process and output results locally allows them to achieve performance comparable to cloud-based post-processing systems (CBPPS). Moreover, local computation addresses key challenges associated with CBPPS, such as high latency, unreliable data transmission, and potential security risks [20]. Anomaly detection on edge devices commonly utilizes Convolutional Neural Networks (CNNs), which have demonstrated strong effectiveness in processing raw vibration signals for fault diagnosis tasks. However, many state-of-the-art CNN architectures are designed for high-performance computing environments and are not well-suited for deployment on resource-constrained embedded devices. In order to address this limitation, a solution involving pruning and reducing the size of a MobileNet-V2 architecture is proposed in [21]. The input time-series data were transformed into spectrogram images using the Constant-Q Nonstationary Gabor Transform, which subsequently used to train the optimized model. Despite achieving reliable results, the combination of the preprocessing step and the model remains computationally intensive. Moreover, the complexity of the preprocessing procedure may limit its compatibility with other lightweight edge devices. A complete framework for real-time bearing fault diagnosis using a Convolutional Neural Network (CNN) deployed on a resource-constrained device is introduced in [22]. The study utilized the Case Western Reserve University (CWRU) bearing dataset and developed a lightweight CNN model optimized through architectural pruning and the use of the Tanh activation function with a ReduceLR adaptive learning rate mechanism. The final model achieved a fault classification accuracy of 98.9% across ten bearing fault types, outperforming conventional CNN baselines in both accuracy and computational efficiency. Although the model was successfully deployed on the STM32

microcontroller using ST's CubeAI framework, the system avoids model quantization, potentially limiting deployment on platforms with stricter memory and computational constraints.

In addition, a system that combines a K-means clustering algorithm with a neural network is implemented on the XIAO ESP32S3 microcontroller as proposed in [23]. Motor vibration signals are captured using an MPU6050 accelerometer and transformed from time domain to frequency domain using Fast Fourier Transform (FFT). The resulting frequency-domain data serves as input for the K-means algorithm to perform anomaly detection. The system achieved an accuracy of 96.5% in identifying anomalies and demonstrated reliable classification performance in real-world conditions. These findings highlight the potential for achieving accurate and efficient anomaly detection on lightweight edge computing platforms.

Despite the progress of recent lightweight and embedded CNN-based bearing diagnosis frameworks, several gaps remain. Most existing methods either depend on complex preprocessing pipelines or avoid full-integer quantization that limit their deployability on ultra-low-power devices. Moreover, many prior studies rely on public datasets (e.g., CWRU) rather than sensor signals collected directly from embedded hardware, resulting in a gap between simulation performance and real-world inference on constrained devices. Therefore, this paper presents a compact and efficient anomaly detection framework for motor shaft and bearing unbalance, designed specifically for deployment on a resource-constrained edge device. Vibration data were collected in-house using the onboard IMU of the Raspberry Pi Pico4ML. These signals were preprocessed through standardization and time-series windowing to prepare them for training and evaluation. A lightweight Convolutional Neural Network (CNN) model was then developed and optimized for size and inference speed. By utilizing an 8-bit quantized architecture with only 2,820 parameters, the proposed system enables real-time and on-device classification without reliance on cloud-based computation. This approach demonstrates the feasibility of implementing accurate and low-latency fault detection within the strict computational and memory constraints of low-power embedded systems, contributing to the advancement of practical edge AI solutions for industrial condition monitoring. The paper is organized as follows: Section 2 provides the equipments and data collection; Section 3 describes the proposed model architecture and its components in detail; Section 4 presents experimental results and comparative analysis; and Section 5 concludes the paper with a summary of findings.

2. **Measurement Setup.**

This study employs a custom-designed rotary testbed for evaluating motor vibration characteristics and implementing edge-based fault detection using deep learning. The measurement configuration is illustrated in Figure 1(a). A single-phase induction motor is used as the primary actuator to rotate a stainless steel shaft. The motor provides a controllable source of rotational motion necessary for generating mechanical vibrations. A Zhengke 4RK25GN-C single-phase induction motor is used as the primary actuator to rotate a stainless steel shaft. A cylindrical shaft is connected to the motor using a flexible coupling and is supported by two pillow block bearings. A 6001-Z shielded deep groove ball bearing is used in this study. It consists of four main components: the outer race, inner race, balls, and cage, as shown in Figure 1(b). The dimensions and specifications are summarized in Table 1. Four distinct bearing conditions are examined in this study: one normal and three faulty states – Outer, Inner Rail, and Cage. The Normal condition represents a healthy, undamaged bearing. The Outer condition involves damage to the outer race, the Inner Rail condition refers to defects on the inner race, and the Cage condition indicates damage affecting both the cage and the ball elements. The visual comparison of bearing conditions used in the experiment is depicted in Figure 2.

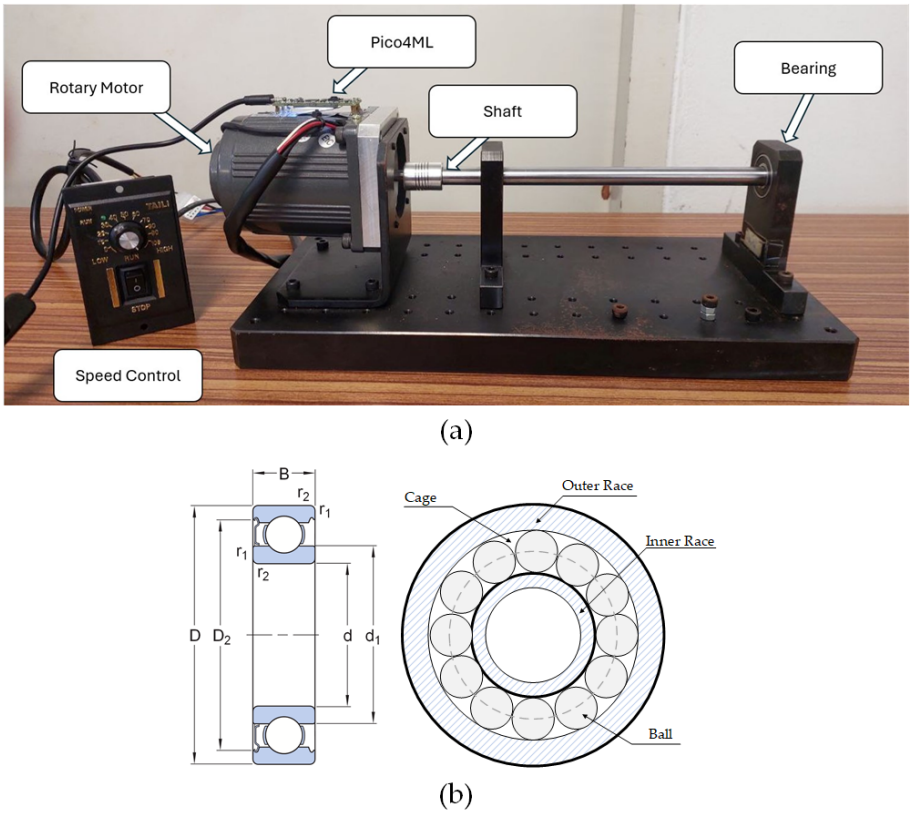


FIGURE 1. (a) Overview of experimental platform and (b) components of bearing

TABLE 1. Bearing parameters

Parameters	Values (mm)
Outside diameter (D)	28
Bore diameter (d)	12
Width (B)	8
Shoulder diameter (d_1)	39.04
Recess diameter (D_2)	24.72
Chamfer dimension ($r_{1,2}$)	min. 0.3
Number of balls	9

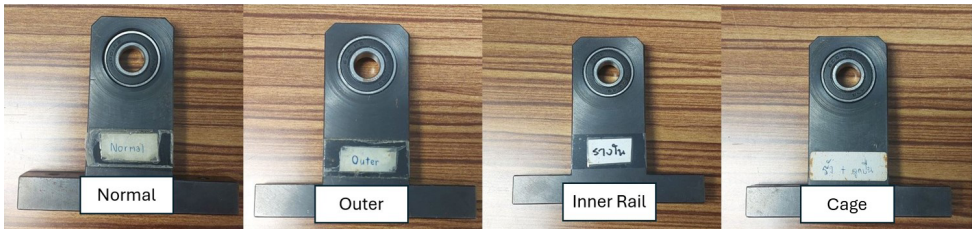


FIGURE 2. Bearing conditions used in the experiment

It is important to note that the performance of the system may be influenced by factors such as sensor mounting position, variations in contact tightness, and external noise sources. These limitations are inherent to IMU-based vibration measurements and should be considered when replicating the experiment or applying the system to different machines.

3. **Proposed CNN-Based Bearing Fault Diagnosis.** The proposed CNN-based bearing fault diagnosis approach involves a systematic two-phase process comprising Data Acquisition and Model Development, as illustrated in Figure 3. Initially, vibration signals indicative of bearing conditions are collected from the onboard Inertial Measurement Unit (IMU). These raw signals undergo a preprocessing stage where standardization and time-series windowing are applied to preparing the data for model training. This preprocessing step enhances the consistency and quality of input data, allowing the CNN model to more effectively discern relevant fault features from the signals. Once preprocessing is completed, the CNN model is trained using the prepared data to classify bearing conditions into normal and various fault types. Following successful model training and validation, the trained CNN model undergoes quantization to reduce its computational complexity and memory footprint, making it suitable for deployment onto resource-constrained hardware. In the final phase, the quantized CNN model is mapped and deployed onto the Raspberry Pi Pico4ML, a low-power edge computing device. During real-time operation, the Pico4ML device continuously processes signals received from the IMU sensor, leveraging the onboard CNN model to diagnose bearing conditions promptly, distinguishing between normal operations and specific fault types. This deployment approach ensures effective and timely fault diagnosis directly at the edge, minimizing response time and improving overall system reliability.

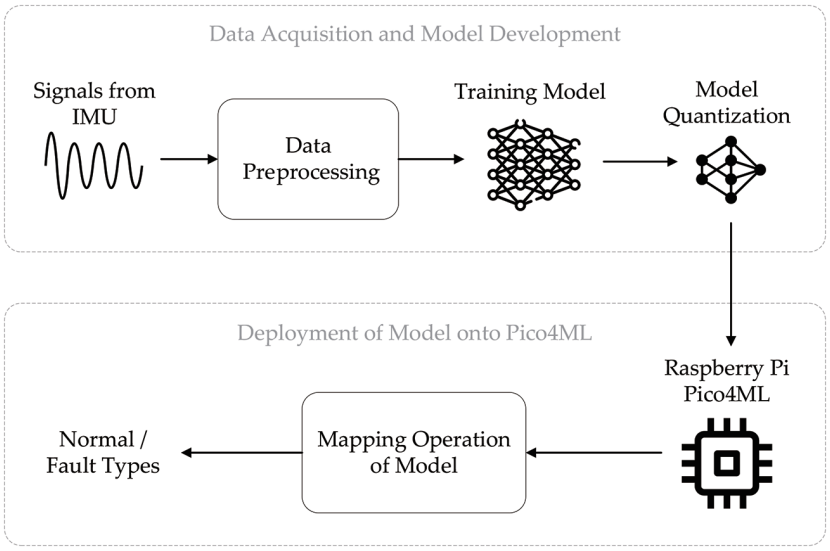


FIGURE 3. Overview of the proposed system

3.1. **Data acquisition and preprocessing.** A Raspberry Pi Pico4ML microcontroller is mounted on top of the rotary motor, which provides a real-time vibration monitoring and classification. The onboard ICM-20948 3-axis accelerometer captures motor surface vibrations, which are critical for detecting subtle anomalies. The captured signals are preprocessed and classified directly on the Pico4ML using a lightweight Convolutional Neural Network (CNN) model specifically trained for this task. The accelerometer captures data at a sampling rate of up to 125 Hz across three axes: X, Y and Z. A built-in digital low-pass filter is employed to attenuate high-frequency noise, thereby preserving essential signal characteristics relevant for anomaly detection. Furthermore, the measurement range is set to 16g, allowing the system to capture a broad spectrum of vibration intensities, which enhances the robustness and sensitivity of the anomaly detection model.

Vibration signals for each bearing condition were recorded over a duration of 30 minutes at a sampling frequency of 125 Hz, which was selected based on the dominant vibration bandwidth of low-speed rotating machinery. This rate sufficiently satisfies the Nyquist criterion while keeping the data size manageable for the memory constraints of the Pico4ML. Due to the limited onboard memory of the Pico4ML, the data is transmitted via serial communication and subsequently stored in an Excel spreadsheet for further analysis. A single IMU was used to maintain low power consumption, minimal wiring, and compatibility with the edge-focused design. For the target application, the overall vibration signature of the shaft and bearing assembly can be reliably captured using a single tri-axial accelerometer, making multi-sensor configurations unnecessary for this study's objectives.

After collecting the four datasets, the vibration signals are preprocessed using standardization and time-series windowing. The Standardization (Z-score normalization) scales the raw input data to have zero mean and unit variance, given by

$$x_{\text{standardized}} = \frac{x - \mu}{\sigma} \tag{1}$$

where x is the original input data point, μ and σ are mean and standard deviation of the time-series data, respectively. As can be seen in Figure 4, standardization normalizes the feature scales across all classes, enhancing the model's ability to extract relevant patterns by minimizing variability. In order to achieve consistency between training and deployment, the calculated mean and standard deviation values must be recorded, as they are essential for real-time inference. Time-series windowing involves segmenting the continuous signal into fixed-length intervals, or "windows", allowing the model to learn localized vibration patterns within each segment. This method improves classification performance and reduces computational overhead during inference. Following the windowing process, each segment is assigned a corresponding class label using one-hot encoding. This technique is well-suited for multiclass classification, as it represents each class with a unique binary vector. One-hot encoding is also compatible with the categorical cross-entropy loss function and aligns with TensorFlow's input requirements for training multiclass classification models.

The combined dataset, consisting of the four labeled classes, is randomly shuffled and divided into three subsets: training (55%), validation (15%), and testing (30%). The training and validation sets are used during model development and tuning, while the

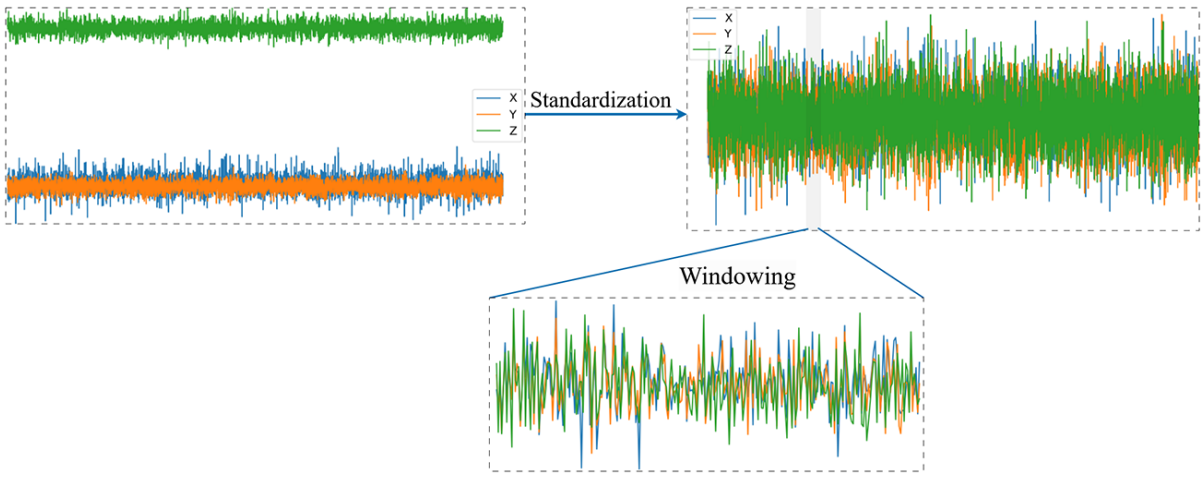


FIGURE 4. Standardization and time-series windowing process

test set is reserved for evaluating the model's performance on previously unseen data. Assessing the model on unseen data is critical for estimating its generalization capability and ensuring reliable performance in real-world deployment scenarios.

3.2. **Model deployment.** The architecture of a custom lightweight CNN model is shown in Figure 5 and detailed in Table 2. The model is specifically designed considering resource constraints to facilitate efficient execution on embedded devices. TensorFlow is employed as the development framework due to its capability to convert models into the TFLite Micro format, ensuring seamless compatibility with the Pico4ML. The model's input shape is $(250, 3, 1)$ where 250 corresponds to the number of time steps, 3 denotes the accelerometer's three-axis measurements, and the final dimension of 1 serves as an additional channel required for compatibility with 2D convolutional (Conv2D) layers. Noted that, this extra channel dimension is introduced because TFLite Micro lacks support for Conv1D operations. The CNN architecture comprises two Conv2D layers, each configured with 16 filters and a kernel size of 3×3. Both layers employ the Rectified Linear Unit (ReLU) activation function to introduce non-linearity, enabling the network to effectively capture complex vibration features. To maintain consistent spatial dimensions throughout the network, particularly crucial given the limited input size, each Conv2D layer utilizes the 'same' padding.

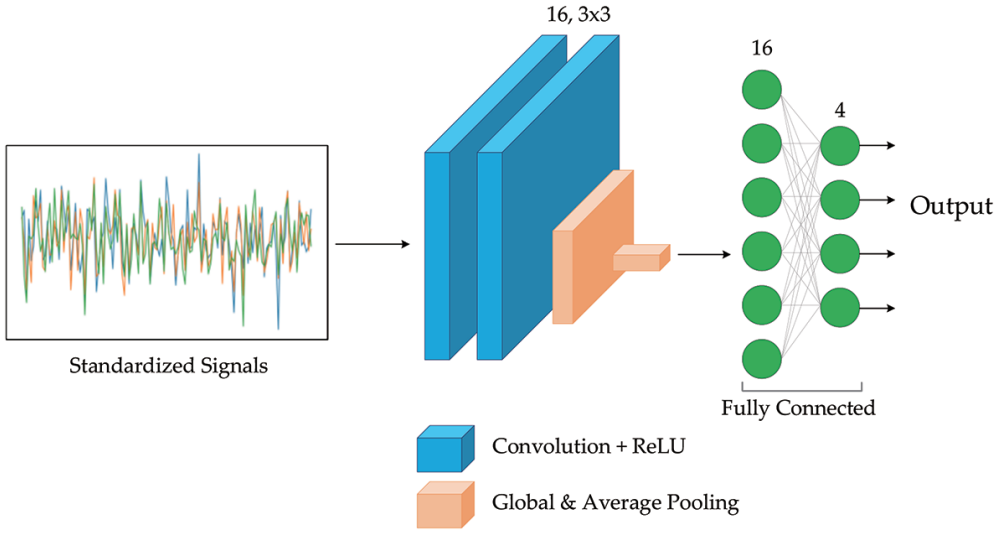


FIGURE 5. Architecture of the proposed lightweight CNN model. The design emphasizes compactness for deployment on resource-constrained devices.

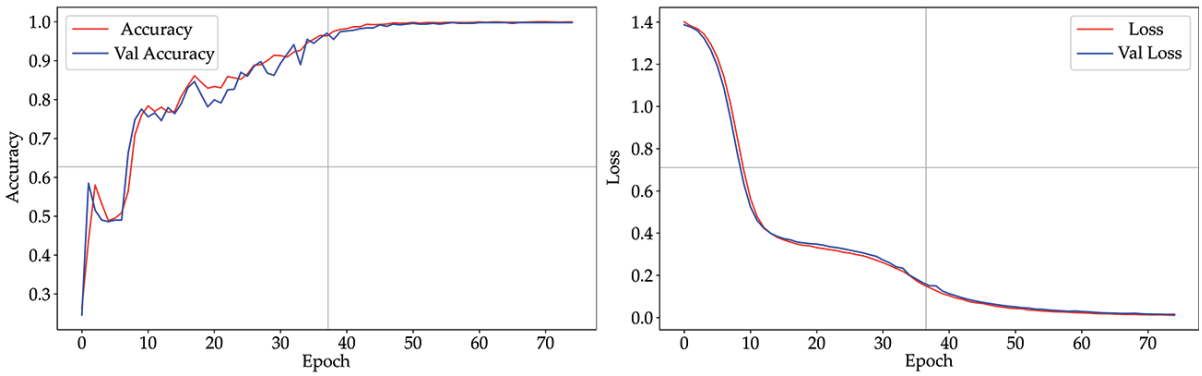
TABLE 2. Model parameters

No.	Layer	Output shape	Parameters
1	Conv2D (conv2d_1)	250, 3, 16	160
2	Conv2D (conv2d_2)	250, 3, 16	2,320
3	AveragePooling2D	125, 1, 16	0
4	GlobalAveragePooling2D	16	0
5	Dense (dense_1)	16	272
6	Dense (dense_2)	4	68
		Total parameters	**2,820**

The use of an AveragePooling2D (AP) layer prior to the GlobalAveragePooling2D (GAP) is found to improve classification performance. By averaging local regions of the feature map, AP smoothens the spatial activations and reduces the local noise, which is particularly useful for models that undergo quantization. Additionally, AP reduces the spatial dimensions of the feature maps, which helps suppress irrelevant background information and enables GAP to focus on the more important features.

GlobalAveragePooling2D (GAP) is employed in place of a conventional flatten layer to substantially reduce the number of trainable parameters, thereby enhancing suitability for deployment on resource-constrained edge devices. GAP operates by computing the average value of each spatial feature map, effectively transforming the output from a shape of $(250, 3, 16)$ into a compact feature vector of 16 dimensions. This architectural choice simplifies the model structure, lowers memory consumption, and enhances generalization performance, which is particularly advantageous when training on limited datasets and aiming to mitigate overfitting. A dense layer with ReLU activation is incorporated before the final classification layer to enhance the model's capacity to learn complex and non-linear representations. Without this intermediate layer, the classification would depend on the linearly aggregated features generated by the GlobalAveragePooling layer, potentially limiting the model's ability to capture subtle inter-class variations.

The final dense layer provides prediction scores as output of the classification. This design enables the application of custom thresholding strategies during deployment, where the scores are evaluated against predefined thresholds to determine the predicted class. Such an approach offers increased flexibility and control over the model's decision-making behavior, facilitating adaptation to specific application requirements. Figure 6 shows the model training progress over 75 epochs. The loss and accuracy curves for both the training and validation datasets indicate effective learning and a good fit, with no evident signs of underfitting or overfitting.



(a) (b)

FIGURE 6. (a) Accuracy and (b) loss of the model during training process

The proposed architecture was selected based on iterative evaluation under embedded constraints. Two Conv2D layers offered the best trade-off between feature extraction capacity and model size; deeper networks increased parameters without meaningful accuracy improvement. The use of AveragePooling before GAP provided additional noise suppression and produced smoother feature maps, which improved performance, particularly after quantization. GAP was preferred over flattening due to its significant reduction in parameters. These components together form a compact yet expressive structure suitable for the Pico4ML.

3.3. **Model quantization.** In order to reduce the model's size and memory usage, the weights and activations are quantized from 32-bit floating-point (float32) representations to 8-bit integer (int8) format using a standard quantization procedure, as follows:

$$float32 = (int8 - zeropoint) \times scale \tag{2}$$

$$scale = \frac{float_{\max} - float_{\min}}{int_{\max} - int_{\min}} \tag{3}$$

During quantization, weights and activations are treated differently. Weights are typically subjected to symmetric quantization, where the *zeropoint* is always set to 0. In contrast, activations can be achieved by asymmetric quantization, allowing the floating-point range to map to any value within the signed int8 interval of $[-128, 127]$. Quantization can be categorized into two types: per-tensor and per-axis. Per-tensor quantization applies a single scale and zero point to the entire tensor, whereas per-axis quantization assigns individual scale and zero point values to each output channel of the weight tensor. The per-axis approach is particularly advantageous for layers like Conv2D, where the multi-dimensional structure of the weights benefits from more granular quantization, thereby enhancing precision and maintaining model accuracy. However, this model intentionally omits per-axis quantization, as it is found to degrade performance and introduce class-wise prediction instability, rather than providing the expected benefits. Although quantization leads to a slight decrease in model performance, the trade-offs are minimal and are outweighed by the reduced inference time and efficiency.

The float32 version of the model occupies approximately 11.2 kB, whereas the fully quantized int8 version requires only 2.82 kB, representing a 75% reduction in memory footprint. This reduction is essential for deployment on microcontrollers with strict RAM and flash constraints. Furthermore, int8 operations are typically executed using simpler integer arithmetic, which is more efficient on embedded processors lacking hardware support for floating-point computation. Quantization therefore not only enables the model to fit within the platform's memory limits but also provides the computational efficiency required for real-time on-device inference.

Finally, the quantized model is converted into a C array format for deployment onto the Pico4ML. Embedding the model as a C array allows it to be compiled into the firmware and directly loaded into memory during execution. The code and data supporting this study are available at https://github.com/amostantan/Faulty-Bearing-Diagnosis-Pico4ML.

4. **Results.** This section presents the experimental evaluation of the proposed model, including classification results, visualizations, and a comparative analysis with existing embedded bearing diagnostic systems. The effectiveness of the proposed CNN-based fault diagnosis model was evaluated using the test dataset comprising unseen vibration signals. Figure 7 depicts the confusion matrix which demonstrates that the model achieves high accuracy in distinguishing among the four bearing conditions: Normal, Outer, Inner Rail, and Cage. The confusion matrix reveals strong diagonal dominance, indicating that the predicted labels closely match the true class labels for most test instances. The details of classification report are also summarized in Table 3.

After deploying the quantized model onto the Raspberry Pi Pico4ML, the model preserved strong real-time inference performance. While some misclassifications were observed between the Inner Rail and Cage classes, particularly when samples from the Cage condition were occasionally predicted as Inner Rail. The overall classification accuracy remained high. This behavior suggests that the internal feature representations for these two classes are relatively similar, posing a greater challenge for separation compared to

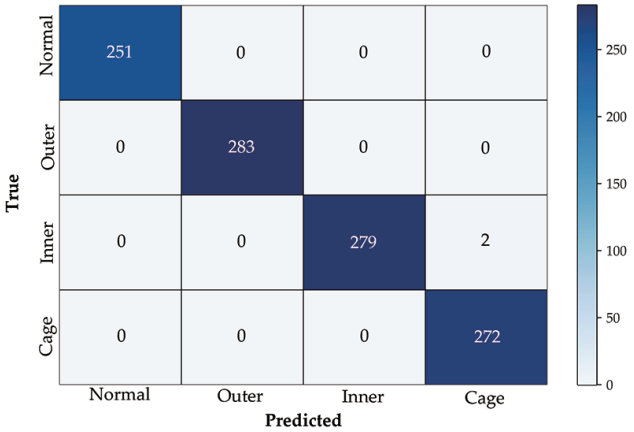


FIGURE 7. Confusion matrix of the CNN model evaluated on the test dataset. The matrix shows the model's ability to distinguish among the four bearing conditions (Normal, Outer, Inner Rail, and Cage), with strong diagonal dominance indicating high classification accuracy.

TABLE 3. Classification report

Classes	Precision	Recall	F1-score	Support
Normal	1.00	1.00	1.00	251
Outer	1.00	1.00	1.00	283
Inner Rail	1.00	0.99	1.00	279
Gun	0.99	1.00	1.00	272
Accuracy			0.9981	1,085

the Normal and Outer classes. Additionally, t-Distributed Stochastic Neighbor Embedding (t-SNE) was applied to the high-dimensional output of the penultimate CNN layer. The two-dimensional visualization, shown in Figure 8, confirms that samples from the Normal and Outer classes form well-separated and compact clusters. Although the Inner Rail and Cage classes appear in close proximity with some degree of overlap, the model is still able to learn subtle feature differences that enable it to distinguish between them effectively in most cases. This observation affirms the discriminative capacity of the proposed CNN architecture, while also indicating potential areas for further refinement, such as enhanced feature extraction or post-processing strategies, to further reduce inter-class confusion. Figure 9 illustrates the activation of the integrated onboard LED when the Pico4ML identifies the Outer fault condition during motor operation. The LED functions as a real-time visual indicator, alerting users to the presence of a bearing fault without requiring additional hardware or external communication. This lightweight feedback mechanism is particularly valuable in resource-constrained environments, enabling immediate fault awareness and facilitating rapid response or shutdown procedures. This demonstration indicates the practical viability of deploying the proposed CNN model on edge hardware for condition monitoring tasks.

A comparative analysis was conducted against several existing embedded bearing fault diagnosis methods, as summarized in Table 4. The comparison includes key metrics such as the embedded platform used, the number of trainable parameters, classification accuracy, and deployment cost. The Proposed System, implemented on the Raspberry Pi Pico4ML, achieves an accuracy of 99.81% with only 2,820 parameters, making it the most lightweight model among the listed methods. This parameter count is significantly lower

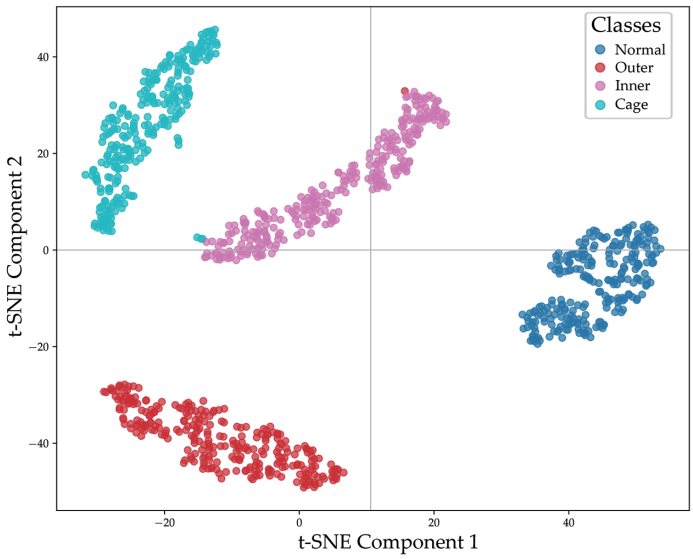


FIGURE 8. t-SNE visualization of the feature representations extracted from the penultimate layer of the proposed CNN model

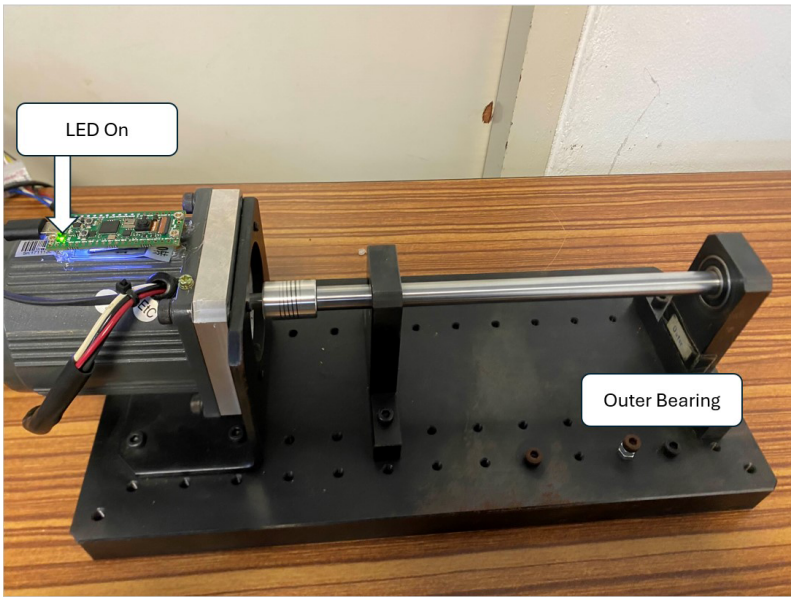


FIGURE 9. LED turns on when a faulty bearing is detected, such as Outer

than other CNN-based approaches, such as the CNN model using the CWRU dataset (36,390 parameters) and the CNN with 28×28 input on ESP32 + ADXL345 (224,131 parameters), while still maintaining competitive classification performance. In terms of accuracy, although some models such as the Param. Transplant CNN (99.84%) and Pruned MobileNet-V2 (99.58%) exhibit slightly higher classification performance, they require substantially more parameters (70,090 and 41,304, respectively), which increases their memory and computation requirements, limiting their suitability for ultra-low-power devices. Moreover, from a cost perspective, the proposed system is also highly efficient, with a total hardware cost of $25.99, which is comparable to or lower than several other platforms. Notably, while the CNN (28×28) method shows high accuracy (99.82%) and lower cost ($21.17), it relies on a significantly more complex model with over 224,000 parameters, which is not ideal for highly constrained environments.

TABLE 4. Comparison with existing systems

Method	Embedded platform	Parameters	Accuracy (%)	Cost (USD)
Param. Transplant CNN [24]	STM32H743	70,090	99.84	$44.4
Pruned MobileNet-V2 [21]	Raspberry Pi 3	41,304	99.58	$30.73
K-means + NN Classifier [23]	XIAO ESP32S3	–	96.5	$25.48
CNN + CWRU Dataset [22]	STM32H743VIT6	36,390	98.9	$21.25
MLP Neural Network [25]	ESP32 + Arduino BLE Sense	–	–	$92.28
CNN (28×28) [26]	ESP32 + ADXL345	224,131	99.82	$21.17
Proposed System	**Raspberry Pi Pico4ML**	**2,820**	**99.81**	**$25.99**

The int8 quantized model achieves an average inference latency of approximately 3 seconds on the Pico4ML. While this delay is longer than typical high-performance embedded platforms, it remains acceptable for low-frequency vibration monitoring, where bearing degradation evolves on the scale of minutes to hours rather than milliseconds. The Pico4ML maintained stable operation within its standard active current range (25-30 mA at 3.3 V), and no observable thermal increase was detected during continuous inference. These observations indicate that despite the modest latency, the proposed method remains viable for low-power, condition-monitoring applications in which real-time streaming inference is not strictly required. Overall, the proposed system demonstrates a compelling balance between model compactness, classification accuracy, and deployment cost, validating its suitability for real-time, low-power bearing fault diagnosis in edge computing environments.

5. **Conclusion.** This paper presents a lightweight and efficient CNN-based approach for bearing fault diagnosis, fully implemented on resource-constrained edge devices. The proposed system effectively processes time-series vibration data acquired from an onboard IMU sensor and achieves accurate classification of four bearing conditions, i.e., Normal, Outer, Inner Rail, and Cage. The designed CNN model can achieve a balance between computational simplicity and classification performance. The model was quantized to 8-bit integer precision and successfully deployed on the Pico4ML, where it maintained robust real-time inference performance. While initial confusion between the Inner Rail and Cage classes was observed, further analysis using t-SNE visualization and logit-based thresholding led to the refinement of decision boundaries, improving classification reliability under deployment conditions. Quantitative evaluation shows that the proposed system achieves 99.81% accuracy with only 2,820 parameters, significantly outperforming or matching existing methods in terms of model compactness and deployment cost. These results demonstrate that reliable bearing condition monitoring can be achieved without reliance on external computation or cloud-based processing, highlighting the practical feasibility of low-cost edge AI solutions for predictive maintenance. The experimental findings further suggest that the proposed framework is well-suited for industrial scenarios where continuous monitoring, low latency, and energy efficiency are essential.

Future work may explore several extensions to enhance the general applicability and robustness of the system. Possible directions include the integration of multiple heterogeneous sensors such as microphones or additional accelerometers, as well as the investigation of adaptive or mixed-precision quantization strategies to reduce latency. Another important step is validating the approach across different machine types and operating environments, where factors such as noise, environmental variability, and long-term drift can significantly influence performance. Advancing these research directions will enable

the proposed method to develop into a more versatile and scalable edge-intelligent monitoring platform.

Acknowledgment. The authors are grateful for the collaboration between Temasek Polytechnic and Thai-Nichi Institute of Technology. The authors would also like to thank Asst. Prof. Pornchai Nivesrangsan and Dr. Watcharin Noothong for the use of their equipment and for their helpful advice.

REFERENCES

[1] C. C. Walani and W. Doorsamy, Edge vs. cloud: Empirical insights into data-driven condition monitoring, *Big Data and Cognitive Computing*, vol.9, no.5, 2025.

[2] X. Tang, L. Xu and G. Chen, Research on the rapid diagnostic method of rolling bearing fault based on cloud-edge collaboration, *Entropy*, vol.24, no.9, 2022.

[3] L. Zuo, L. Zhang, Z.-H. Zhang, X.-L. Luo and Y. Liu, A spiking neural network-based approach to bearing fault diagnosis, *Journal of Manufacturing Systems*, vol.61, pp.714-724, 2021.

[4] U. Inyang, I. Petrunin and I. Jennions, Health condition estimation of bearings with multiple faults by a composite learning-based approach, *Sensors (Basel)*, vol.21, no.13, 2021.

[5] S. Zhang, S. Zhang, B. Wang and T. G. Habetler, Deep learning algorithms for bearing fault diagnostics – A comprehensive review, *IEEE Access*, vol.8, pp.29857-29881, 2020.

[6] MuthuKumaran M, Reema Mansoora S, Shamna B S, Suryavadhani D and Hariraj R, Intelligent predictive maintenance for motor health monitoring using IoT sensors and machine learning algorithms, *2025 8th International Conference on Trends in Electronics and Informatics (ICOEI)*, pp.594-600, 2025.

[7] L. Wan, K. Gong, G. Zhang, X. Yuan, C. Li and X. Deng, An efficient rolling bearing fault diagnosis method based on spark and improved random forest algorithm, *IEEE Access*, vol.9, pp.37866-37882, 2021.

[8] D. Goyal, A. Choudhary, B. Pabla and S. S. Dhami, Support vector machines based non-contact fault diagnosis system for bearings, *J. Intell. Manuf.*, vol.31, 2020.

[9] B. Sun and X. Liu, Significance support vector machine for high-speed train bearing fault diagnosis, *IEEE Sensors Journal*, vol.23, no.5, pp.4638-4646, 2023.

[10] A. Sharma, R. Jigyasu, L. Mathew and S. Chatterji, Bearing fault diagnosis using weighted k-nearest neighbor, *2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI)*, pp.1132-1137, 2018.

[11] M. A. Jamil, M. A. A. Khan and S. Khanam, Feature-based performance of SVM and KNN classifiers for diagnosis of rolling element bearing faults, *Vibroengineering PROCEDIA*, pp.36-42, 2021.

[12] S. Lu, G. Qian, Q. He, F. Liu, Y. Liu and Q. Wang, In situ motor fault diagnosis using enhanced convolutional neural network in an embedded system, *IEEE Sensors Journal*, vol.20, no.15, pp.8287-8296, 2020.

[13] Z. Guo, M. Yang and X. Huang, Bearing fault diagnosis based on speed signal and cnn model, *Energy Reports*, vol.8, pp.904-913, 2022.

[14] Q. Cui, Z. Li, J. Yang and B. Liang, Rolling bearing fault prognosis using recurrent neural network, *2017 29th Chinese Control and Decision Conference (CCDC)*, pp.1196-1201, 2017.

[15] S. Eleftheriadis, G. Margaritis, C. S. Kouzinopoulos, D. Ioannidis and D. Tzovaras, An embedded system for the predictive maintenance of aerobic digesters with liquid outputs, *2024 13th International Conference on Modern Circuits and Systems Technologies (MOCAST)*, pp.1-4, 2024.

[16] W. Wang, J. Yu, Y. Ma, Z. Pan and T. Chen, Bearing fault diagnosis based on deep learning and array stochastic resonance under strong noise background, *International Journal of Innovative Computing, Information and Control*, vol.21, no.2, pp.549-563, 2025.

[17] C. Janiesch, P. Zschech and K. Heinrich, Machine learning and deep learning, *Electronic Markets*, vol.31, pp.685-695, 2021.

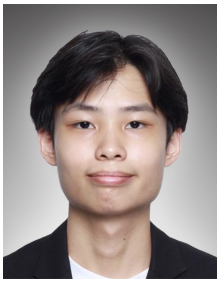
[18] L. C. Brito, G. A. Susto, J. N. Brito and M. A. V. Duarte, Fault detection of bearing: An unsupervised machine learning approach exploiting feature extraction and dimensionality reduction, *Informatics*, vol.8, no.4, 2021.

[19] S. Leroux and P. Simoens, Hybrid edge-cloud models for bearing failure detection in a fleet of machines, *Electronics*, vol.13, no.24, 2024.

[20] M. P. Véstias, A survey of convolutional neural networks on edge with reconfigurable computing, *Algorithms*, vol.12, no.8, 2019.

[21] M. T. Pham, J.-M. Kim and C. H. Kim, Deep learning-based bearing fault diagnosis method for embedded systems, *Sensors*, vol.20, no.23, 2020.
[22] W. Liao, Real time bearing fault diagnosis based on convolutional neural network and STM32 microcontroller, *arXiv Preprint*, arXiv: 2304.09100, 2023.
[23] S. Arciniegas, D. Rivero, J. Piñan et al., IoT device for detecting abnormal vibrations in motors using TinyML, *Discover Internet of Things*, vol.5, p.41, 2025.
[24] X. Ding, H. Wang, Z. Cao, X. Liu, Y. Liu and Z. Huang, An edge intelligent method for bearing fault diagnosis based on a parameter transplantation convolutional neural network, *Electronics*, vol.12, no.8, 2023.
[25] A. F. Cotrino Herrera, J. A. López Sotelo, J. C. Blandón Andrade and A. T. Lazo, Low-cost prototype for bearing failure detection using Tiny ML through vibration analysis, *HardwareX*, vol.22, e00658, 2025.
[26] A. W. Lucas, M. P. d. S. Caique, R. D. Dantas, S. Vanessa and C. d. S. J. Wilson, Bearing fault detection via convolutional neural networks using low-cost MEMS accelerometers, *Revista Caleidoscópio*, vol.16, no.1, 2024.

Author Biography



Amos Tan is currently a student in Computer Engineering at Temasek Polytechnic, Singapore. He contributed to the research as part of an internship collaboration between Temasek Polytechnic (Singapore) and Thai-Nichi Institution of Technology (Thailand) in April 2025 to August 2025. His academic interest includes IoT, data analytics and Edge AI.

Chatchai Wannaboon received the B.Eng. degree in Computer Engineering from Thai-Nichi Institute of Technology, Thailand, 2013; the M.Eng. degree in Electronic and Photonic System Engineering from Kochi University of Technology, Japan, 2015; the Ph.D. degree in Electronic and Photonic System Engineering, from Kochi University of Technology, Japan, 2018.

Dr. Wannaboon is currently a full-time professor at the Computer Engineering and Artificial Intelligence, Thai-Nichi Institute of Technology, Thailand. His main research interests include the applied machine/deep learning, edge computing, and system integration.