Proceedings of the 2022 International Symposium on Flexible Automation
ISFA2022
July 3 - 7, 2022, Yokohama, Japan

ISFA2022-C070

IIoT Deployment of a Physics-Informed Deep Learning Model for Online Bearing Fault Diagnostics

Hao Lu, Cade Allen, Venkat Nemani, Chao Hu[1]
Department of Mechanical Engineering
Iowa State University
Ames, IA

Andrew Zimmerman
Percēv LLC and Grace Technologies
Davenport, IA

ABSTRACT

Early fault detection in rolling element bearings is pivotal for the effective predictive maintenance of rotating machinery. Deep Learning (DL) methods have been widely studied for vibration-based bearing fault diagnostics largely because of their capability to automatically extract fault-related features from raw or processed vibration data. Although most DL models in the current literature can provide fairly accurate classification outputs, the typical diagnostic procedure is performed in an offline environment utilizing powerful computers. This centralized approach can lead to unacceptable delays in safety-critical applications and can prohibit cost-sensitive wireless data collection. Meanwhile, very few studies have reported on deploying DL models on microprocessor-based Industrial Internet of Things (IIoT) devices, where edge computing can give users a real-time evaluation of bearing health without requiring expensive computational infrastructure. This paper demonstrates an IIoT deployment of a physics-informed DL model inside a commercially available wireless vibration sensor for online health classification. The diagnostic model here is developed and trained offline, and the trained model is then deployed inside the embedded system for online prediction. We demonstrate the model's online diagnostic performance by imitating bearing vibration signals on a vibration shaker and by performing edge computing on the embedded system mounted on the shaker.

Keywords: Deep learning, bearing health monitoring, edge computing, embedded system.

1. INTRODUCTION

Modern industrial equipment is becoming increasingly complex due to recent advances in automation. As a result, the necessity of equipment health monitoring has increased over the years. Because bearing failures are considered to be the primary cause of degradation in rotating machines (pumps, motors, compressors, etc.), detecting early-stage bearing faults in a timely manner would help maintenance teams ensure the safe operation of critical rotating machinery without unexpected breakdown [1].

There has been extensive research in the field of bearing diagnostics, where bearing fault detection is generally conducted in three steps: data acquisition, feature extraction, and pattern recognition [2]. The data acquisition step uses sensors to collect signals that are affected by bearing health. Different sensors have been applied for bearing diagnostics, including acoustic, vibration, and temperature sensors. The feature extraction step extracts fault-related features from the raw sensor data. Signal processing techniques such as Fourier transform, wavelet transform, and short-time Fourier transform are commonly used for extracting fault-related features from the raw data. The pattern recognition step uses the extracted features to predict bearing health conditions with the help of various machine learning or deep learning (DL) techniques. Over the past few years, researchers have tested a number of DL models and reported decent accuracy using pre-collected test data [3]. However, most of these models are built and evaluated in an offline manner, and very few researchers reported DL models for online bearing fault diagnostics.

In the era of the Industrial Internet of Things (IIoT), a traditional approach to implementing the diagnostic model is to use a local device to collect data and upload that data to the cloud. The DL model, deployed in the cloud, then makes predictions and sends the results back to the local device [4]. However, transmitting raw vibration data to the cloud increases power consumption, affecting the battery's useful life and increasing operating costs [5]. As a potential solution to these limitations, edge computing has become an important technique for IIoT services, where an embedded system performs diagnostic tasks locally at the data source. The advantages of edge computing are not limited to reducing data transmission costs, as it can also provide real-time evaluation results and preserve a user's data privacy.

This paper presents an IIoT deployment of a physics-informed DL model for online bearing health diagnostics. We incorporate physical knowledge of bearing failure to extract fault-related vibration features and develop a memory-efficient

[1] Contact author: chaohu@iastate.edu, huchaostu@gmail.com

algorithm for IIoT implementations. The key contributions are as follows:

- Deploy a signal processing and DL algorithm into an embedded system.
- Evaluate the performance of the embedded diagnostic algorithm and compare its prediction accuracy with a traditional offline PC model developed in Python using the Numpy and Keras packages [6].

2. BACKGROUND

When a bearing with a defect rotates, the contact between the defect area and other bearing components causes vibration impulses. These bearing fault impulses also excite other bearing components, leading to vibration modulation. Envelope analysis is used to demodulate the vibration signal and extract low-frequency fault-related signals. The frequencies of the fault-related signals, also named bearing fault characteristic frequencies, are related to the bearing's geometry. Further details of Hilbert transform-based demodulation and extraction of physics-based features can be found in [2].

In Ref. [7], we proposed a physics-informed DL architecture, which uses a feature weighting layer that assigns higher weights to feature values close to the bearing fault characteristic frequencies. We then verified that physics-informed feature weighting helps improve the model's sensitivity in identifying the bearing faults and provides better diagnostic accuracy. This paper demonstrates the deployment of the same physics-informed convolutional neural network (PICNN) inside an embedded system.

3. METHODOLOGY

The architecture of the hardware platform is shown in Figure 1. This embedded system comprises two microprocessors (ATSAMG55J19 and ATxmega128A1U) and an accelerometer (KX131). The signal processing algorithm and DL model are converted from Python into C and compiled into the ATSAMG55J19, which can access the accelerometer. The ATSAMG55J19 performs signal processing and diagnostics, then sends the envelope order spectrum and health class results to the ATxmega chip. The ATxmega chip is then used to transmit the data to a server.

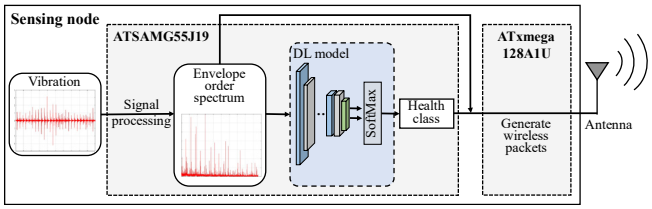


Figure 1. The architecture of the diagnostic system

The deployment of signal processing and DL algorithms into an embedded system requires considering both memory usage and computational complexity [8]. The signal processing section contains three algorithms: Hilbert transform, Fourier transform, and interpolation. The Hilbert transform could be performed with the help of the Fourier transform [9]. To reduce computational complexity, we use the fast Fourier transform (FFT) algorithm; compared to the Fourier transform, the FFT reduces the computational complexity from $O(n^2)$ into $O(n \log(n))$ [10]. Regarding the DL model, saving all the model parameters and outputs of each layer requires too much memory. The memory allocation for the embedded system can be treated as an optimization problem. Taking a four-layer DL model as an example, Figure 2(a) illustrates a naive way of memory allocation over time. One dimension is the memory size, and the other dimension is the time during which each memory allocation must be preserved. For example, the orange rectangular "A" represents the input and the parameters for the signal processing operation, and "B" represents the output of signal processing. In this paper, the bin packing memory allocation strategy is adopted to optimize memory usage, as shown in Figure 2(b). This strategy places each allocation in the first sufficiently large gap or at the end of the buffer if the gap is insufficient for memory allocation. After the results of the current layer are calculated, the input of this layer is erased from the memory, and the erased memory is assigned for the output of the next layer.

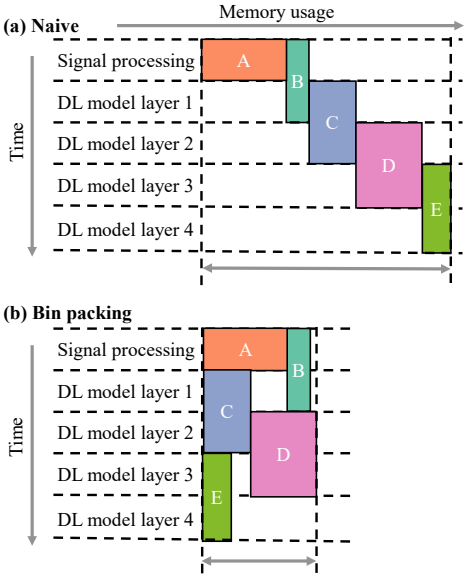
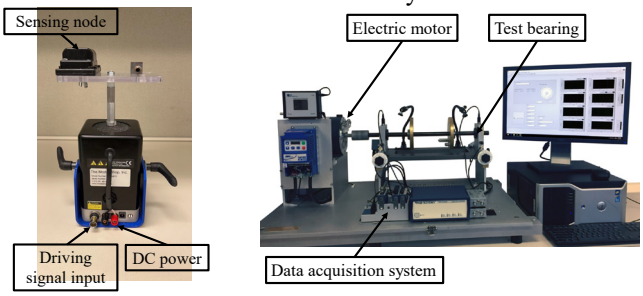


Figure 2. Memory allocation strategies

The implementation of the diagnostic model on the wireless sensing node is composed of four steps. First, the DL model is trained on a PC using Keras, then the model architecture and model parameters are converted into a C code. Then, the C code is compiled into the sensing node, and finally, we perform online experiments to verify model performance.

4. EXPERIMENTAL VERIFICATION

Following our earlier work [7], we train two models, a PICNN and a CNN, and then deploy them into the embedded system separately. Training data is collected from the bearing fault simulator, shown in Figure 3b. Two bearings are mounted on the shaft of the simulator and driven by an electric motor. The



(a) (b)

Figure 3. (a) Shaker test stand and (b) bearing fault simulator

accelerometer (PCB 608A11) is mounted on the bearing housing to gather vibration signals.

We use bearings with (1) an inner race fault, (2) an outer race fault, and (3) a combination of faults, which have been introduced previously by electrical discharge machining. The test bearings are rolling element bearings with 13 balls with the inner race, outer race, and ball diameters as 22.1 mm, 29.1 mm, and 3.5 mm, respectively. The vibration data is collected under shaft speeds ranging from 15 Hz to 30 Hz. A total of 15 vibration signals are collected from each bearing at a sampling rate = 12.8 kHz for 20 s. The sliding window signal is divided into multiple samples of 1 s per sample with an overlap of 0.5 s. In total, the training dataset contains 570 samples (Table 1).

Table 1. Dataset summary

	Parameter	Value
Training dataset	Shaft speed (Hz)	15, 16, 17, …, 30
	Bearing condition	Healthy, inner race fault, outer race fault, combination of faults
Shaker test	Shaft speed (Hz)	15.5, 20.5, 25.5, 30.5
	Bearing condition	Healthy, inner race fault, outer race fault, combination of faults

The architecture of the PICNN is included in Table 2. The architectural difference between the CNN and PICNN is that the CNN does not have the feature weighting layer. The feature weighting layer, which is the first layer of the PICNN, assigns higher weights to the features close to the bearing fault characteristic frequencies.

Table 2. Architecture of PICNN

Layer name	Output shape, Activation
Feature weighting	(Samples, 1600,1)
Conv-1	(Samples,793,8)
Average-pool	(Samples,198,8)
Conv-2	(Samples,96,16)
Average-pooling	(Samples,32,16)
Conv-3	(Samples,14,32)
Average-pooling	(Samples,7,32)
DC-1	(Samples,4)

Although the signal processing and DL-based diagnostic algorithms have been verified on a PC, it is essential to confirm that the model can provide identical results after being deployed into the embedded system. Thus, a shaker test is conducted where an electrodynamic vibration shaker converts voltage signals into vibration; with a proportional control algorithm, the shaker can replicate the vibration signals collected from the bearing test stand. The sensing node is mounted on the shaker to perform online diagnostics of the vibration signals.

Unlike typical offline tests where pre-collected samples are used to evaluate the model's accuracy, the sensing node here performs online diagnostics by classifying bearing health immediately after collecting the data. For each experimental setting, the wireless sensing node performs five diagnostic evaluations. Then, the diagnostic results and the processed envelope spectrum are sent to the server. The server with a wireless receiver receives the data and performs its own diagnostic evaluation using the Keras-based DL model and the envelope spectrum. All the test results are summarized in Table 3. Although test data was collected using different shaft speeds, both the PICNN and CNN provide accuracy greater than 90%. Also, the results provided by the embedded system are identical to those by the PC model, indicating that the diagnostic models are successfully deployed in the embedded system.

Table 3. Diagnostic results

Healthy | Inner race fault | Outer race fault | Combination of faults

Item		Embedded model results								Keras model results							
		CNN				PICNN				CNN				PICNN			
True label		20	0	0	0	20	0	0	0	20	0	0	0	20	0	0	0
		0	17	0	3	0	18	0	2	0	17	0	3	0	18	0	2
		0	0	20	0	0	0	20	0	0	0	20	0	0	0	20	0
		0	2	0	18	0	1	0	19	0	2	0	18	0	1	0	19
Accuracy		93.75%				**96.25%**				93.75%				**96.25%**			

5. CONCLUSION

This study proposes a methodology for deploying a DL model into an embedded system for online bearing fault diagnostics. Using a shaker test, we validate that the embedded model provides identical results to the PC model. Our future work will investigate the applicability of the proposed embedded model for online diagnostics using the bearing fault simulator.

ACKNOWLEDGEMENTS

This work was supported in part by the U.S. National Science Foundation under Grants IIP-1919265 and IIP-2036044. Any opinions, findings, or conclusions in this paper are those of the authors and do not necessarily reflect the sponsor's views.

REFERENCES

[1] Randall, R.B. and J. Antoni, *Rolling element bearing diagnostics—A tutorial.* Mechanical systems and signal processing, 2011. **25**(2): p. 485-520.

[2] Shen, S., et al., *A physics-informed deep learning approach for bearing fault detection.* Engineering Applications of Artificial Intelligence, 2021. **103**: p. 104295.

[3] Hoang, D.-T. and H.-J. Kang, *A survey on deep learning based bearing fault diagnosis.* Neurocomputing, 2019. **335**: p. 327-335.

[4] Yang, S., et al., *A unified framework and platform for designing of cloud-based machine health monitoring and manufacturing systems.* Journal of Manufacturing Science and Engineering, 2015. **137**(4).

[5] Azar, J., et al., *An energy efficient IoT data compression approach for edge machine learning.* Future Generation Computer Systems, 2019. **96**: p. 168-175.

[6] Gulli, A. and S. Pal, *Deep learning with Keras*. 2017: Packt Publishing Ltd.

[7] LU, H., et al., *Fault Diagnosis of Rolling Element Bearings on Low-Cost and Scalable IIoT Platform.* Structural Health Monitoring 2019, 2019.

[8] David, R., et al., *TensorFlow lite micro: Embedded machine learning on tinyml systems. arXiv 2020.* arXiv preprint arXiv:2010.08678.

[9] Liflyand, E., *Interaction between the Fourier transform and the Hilbert transform.* Acta et Commentationes Universitatis Tartuensis de Mathematica, 2014. **18**(1): p. 19-32.

[10] Lin, S., et al. *FFT-based deep learning deployment in embedded systems*. in *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. 2018. IEEE.