Fault Diagnosis for Industrial Rotary Machinery based on Edge Computing and Neural Networking

Valentin Perminov*, Vladislav Ermakov*, Dmitry Korzun*
* Petrozavodsk State University (PetrSU), Petrozavodsk, Russia
Email: perminov@cs.petrsu.ru, vlaermak@cs.petrsu.ru, dkorzun@cs.karelia.ru

***Abstract*—Recent progress in Sensorics and Internet of Things (IoT) enables real-time data analytics based on data from multiple sensors covering the target industrial production system and its manufacturing processes. Diagnostics and prognosis can be implemented using the neural network approach on top of vibration and other sensed data. Neural network methods lead to high accuracy in fault detection and fault evolution. Nevertheless, transferring a neural network model to edge devices leads to performance issues and platform limitations. In this paper, we discuss the edge computing opportunities for diagnostics of industrial rotary machinery using well-known neural network methods.**

Keywords–Fault diagnosis; convolutional neural network; edge computing; vibration diagnostics.

I. INTRODUCTION

The recent progress in Sensorics and Internet of Things (IoT) enables real-time data analytics for industrial systems [1]. The analytics—diagnostics and prognosis—is based on data coming in from multiple sensors [2] (i.e., multi-parametric monitoring). Many sensors cover the target industrial production system to monitor its technical state, utilization conditions, and underlying manufacturing processes.

In this paper, we focus on the fault diagnostics problem for industrial rotary machinery. Mechanical parts (e.g., rolling bearings and electric motor rotors) are monitored in real-time [3]. First, we study the problem of applying the Convolutional Neural Network (CNN) methods to detect faults and to evaluate fault characteristics [4]. Second, we study how low-capacity edge IoT devices can be useful to perform data analysis in real-time [5].

Diagnostics and prognosis can be implemented using the CNN methods on top of vibration and other sensed data [6]. The CNN methods lead to high accuracy in fault detection and evaluation [7]. The topical practical problem in industrial rotary machinery fault diagnosis is bearing fault detection. The diagnostics could be done by analysis of vibration data from sensors installed at the unit under monitoring. We consider the case when a CNN is applied for bearing fault classification, based on raw vibration signal analysis.

To obtain diagnostics results in real-time the network latency and traffic volume should be reduced in transferring data to the processing center. Data processing must be implemented near the machinery in real-time. We consider an industrial monitoring system with edge CNN computing devices. Flow data from the sensors go to an edge CNN computing device either directly through wired/wireless interfaces or through aggregator IoT devices. Deploying a CNN model to edge devices has platform limitations and leads to the performance issues. We experiment with the performance estimation of such edge device. We show that low-capacity edge IoT devices have enough performance to make data analysis based on CNN in industrial monitoring tasks.

The rest of the paper is organized as follows. Section II considers existing approaches to rotating machinery fault diagnosis: Condition Monitoring (CM) and Prognostic Health Monitoring (PHM). Section III defines the problem of vibration diagnostics in rotary machinery. Section IV shows the possible methods to solve the problem of vibration diagnostics in rotary machinery. Section V presents positive results of our feasibility study on the applicability. Section VI summarizes the results of this study.

II. RELATED WORK

There are two approaches to rotating machinery fault diagnosis: CM and PHM.

The first technique is a condition monitoring using feature extraction from the raw signal. In the study [3], there is a method for CM based on feature extraction from the raw signal. The raw signal is divided into frames, and the set of features from each frame is extracted. The feature set for each frame includes time, frequency, and time-frequency domain features. The extracted features are used for machine learning algorithms to classify the rolling bearing condition state.

In [8] Remaining Useful Life (RUL) criterion is proposed to estimate bearing condition in the future. The proposed method includes filtering the raw signals using Discrete Wavelet Transform (DWT), then extracting time, frequency, and time-frequency domain features. An autoregressive model is then established for each of the extracted features. The optimum features are then selected using a kernel-based Extreme Learning Machine (ELM) algorithm and Performance Evaluation Criterion (PEC). In the training stage of the method, the selected features are used as inputs to the PHM algorithm, while the RUL is used as the target vector. A degradation model is obtained after the training stage, which is used together with the testing input features to predict the fractional RUL of the test data.

Another approach is a raw signal analysis using Deep Learning (DL) algorithms. In [6], CNN is used to evaluate the rolling bearing fault type. The vibration signal presented as a vector of N-samples is selected as an input for CNN. Plenty of convolution layers extracts features from the raw signal that is analyzed by fully-connected layers. The output of the proposed model presents an M-length vector, where M — is the number of condition states. Additionally, data fusing is applied to evaluate more performance from the proposed method. In this case, the input data presented as a set of vectors: vibration signal vector, and two-phase motor current vector, that spins the shaft with installed bearing.

Palossi et al. demonstrated a navigation engine for autonomous nano-drones capable of closed-loop end-to-end

 ISBN: 978-1-61208-811-2

CNN-based visual navigation [9]. They deployed DroNet neural network to the GAP8 processor and achieved power consumption of CNN processing of only 64 mW on average.

In [10], a real-time fault detection system called LiReD was implemented for an industrial robot manipulator. The system consisted of a Raspberry Pi single-board computer and a piezoelectric accelerometer. The Long Short-Term Memory (LSTM) recurrent neural network was used to analyze the vibration signal with the aim of vacuum ejector fault detection. The LSTM-based fault detection model was compared with k-Nearest Neighbor with Dynamic Time Warping (k-NN+DTW), Random Forest (RF), and Support Vector Machine (SVM). The LSTM-based fault detection model showed the best performance, among others. The authors note that more complex analysis and more complex neural networks will require more efficient hardware, and retraining and compression techniques that reduce the size of the model but increase or maintain its performance.

III. Vibration Diagnostics in Rotary Machinery

To keep industrial machinery in appropriate condition, the methods of technical condition monitoring (diagnostics) and predictive maintenance are applied. The diagnostics are based on the current machinery state by obtaining signals from various sensors. The predictive maintenance aims at forecasting behavior mechanisms at a certain point in time with the current state. Both of these methods found application in the Industrial Internet of Things (IIoT) diagnostic systems. The utilization of condition diagnostics and predictive maintenance services establishes an effective equipment machinery operation mode and personnel timetable.

A. Condition monitoring for rolling bearing

Rotary machinery and their mechanical parts, such as bearings and electric motor rotors, must be monitored online. CM with offline-based systems performs post-processing operations on a remote server. Hence the results of the diagnostics could not be obtained instantly. To get this result near the machinery online, the special methods and hardware should be applied. This possibility could be obtained by using edge computing devices. These devices are portable apparatus installed near the machinery for online condition diagnostics. The raw data from various sensors, installed on an object, are collected by the edge device for analysis. The flow-based data, in case of continuous sensing and processing of different data types, are performed for:

- mechanical parts vibration, position, speed;
- electric motor current;
- temperature;
- acoustic signals.

The methods need to analyze data from various sensors by applying special techniques to increase performance on an edge computing device. Especially for bearings, the condition state could be obtained by a vibration signal analysis. Some approaches include envelope spectrum analysis with Fast Fourier Transform (FFT) and neural network methods with feature extraction. The vibration signal and its spectrum includes the most important information about bearing condition state. For example, the vibration signal, in case of inner ring defect, presents a normal noise, modulated by hight order rotary speed harmonics. When developing a mobile device for CM, the selected hardware must provide an effective implementation of the diagnostic method through the use of dedicated hardware processing units, such as Digital Signal Processing (DSP) unit and neural network accelerator. Hence, in this work, edge computing opportunities were applied to rolling bearing diagnosticians with the proposed model based on CNN.

B. The dataset description

The big dataset should be used to train a CNN. For online condition diagnostic, it is important to know a bearing state: ”healthy” or ”damaged”. The causes and types of damages could be studied with server-class computers. To evaluate the applied model performance, the Paderborn University Bearing Dataset was selected as a dataset for train and test data [3]. The dataset consists of MatLab files with measurements from various sensors, such as accelerometer, current sensor, torque sensor, and thermometer. The experimental dataset was obtained using a specific test rig, see Figure 1, which was designed and operated at the Chair of Design and Drive Technology, Paderborn University.

The test rig is a modular system to ensure the flexible use of different defects in an electrically driven mechanical drive train. The test rig consists of several modules: an electric motor (1), a torque-measurement shaft (2), a rolling bearing test module (3), a flywheel (4), and a load motor (5).

The ball bearings with different types of damage are mounted in the bearing test module to generate the experimental data. For the generation of the measurement data, the current signals of the electric motor are recorded. The vibration and current signal were measured with a 64 kHz sample rate under different conditions for 32 bearings with four-digit code.

The dataset provides four types of bearings, described in Table I: “healthy” – with no damages, “IR” – inner ring defect, “OR” – outer ring defect, “IR+OR” – both inner and outer ring defects.

TABLE I. The dataset bearings.

Healthy	*”IR”*	*”OR”*	*”IR+OR”*
K001	KA01	KI01	KB23
K002	KA03	KI03	KB24
K003	KA04	KI04	KB27
K004	KA05	KI05	
K005	KA06	KI07	
K006	KA07	KI08	
	KA08	KI14	
	KA09	KI16	
	KA15	KI17	
	KA16	KI18	
	KA22	KI21	
	KA30		

According to the nature of the defect, damaged bearings belong to two main groups, which are described in Table II: artificially damaged bearings and bearings with real damages. Artificial damages were made by electric discharge machining, drilling, and manual electric engraving with a different extent. Real damages were generated by accelerated lifetime tests.

Each bearing used to run under different speed, torque and radial load — 20 measurements of 4 seconds each for each operating condition, saved as a MatLab file (80 in total) with a name consisting of the code of the operating condition and the four-digit, bearing code (e.g., N15_M07_F10_KA01_1.mat).

IV. Edge-Centric Neural Network Computing

In this section, we suggest the concept of an industrial monitoring system and describe its prototype used in this

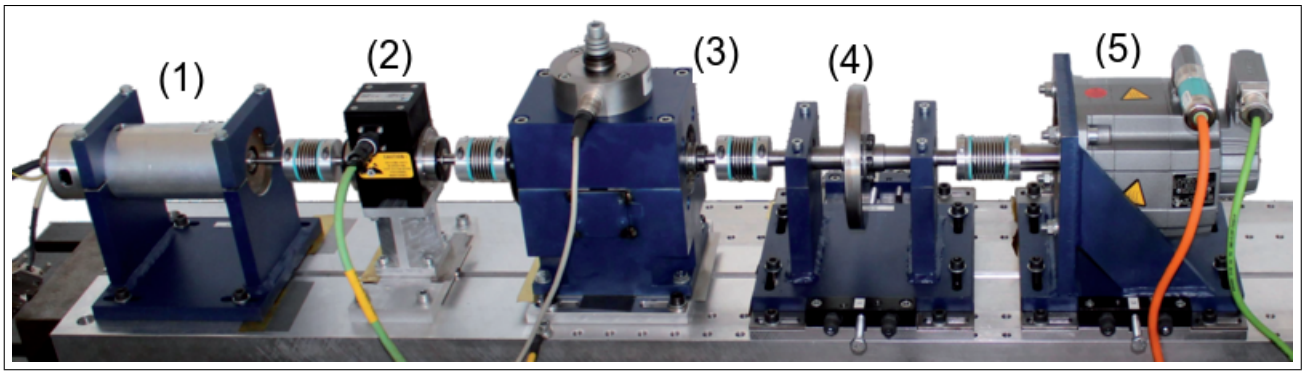

Figure 1. Test rig setup: (1) – electric motor, (2) – torque-measurement shaft, (3) – rolling bearing test module, (4) – flywheel, (5) – load motor [3].

TABLE II. DAMAGED BEARINGS TYPES.

Artificially damaged	*With real damages*
KA01	KA04
KA03	KA15
KA05	KA16
KA06	KA22
KA07	KA30
KA08	KB23
KA09	KB24
KI01	KB27
KI03	KI04
KI05	KI14
KI07	KI16
KI08	KI17
	KI18
	KI21

paper. Also, we provide a description of CNN for bearings fault detection, training and validation datasets, and used software tools and frameworks.

A. Edge-Centric Neural Network Computing Device

The rotating machinery fault diagnosis as a part of the industrial monitoring system could extract information from various sensors to make desitions about technical condition of equipment. Such sensors could be accelerometers, encoders, thermosensors, current and acoustic sensors. The data from sensors could flow to neural network computing device directly through wired or wireless interfaces or through aggregation devices. In our experiment, we simulate dataflow from sensors by a personal computer reading files from bearing vibration signal dataset and send frames of this signal via Universal Asynchronous Receiver-Transmitter (UART) to the neural network computing device. As such a device, we use Kendryte K210 system-on-chip, which has build-in dual-core Central Processing Unit (CPU) to execute a control algorithm, peripheral interfaces to interact with different sensors and communication modules, including digital accelerometers and wireless adapters, and CNN hardware accelerator unit designed for efficient CNN inference [11]. These properties make it well suited for IoT applications. And as we show in Section V, its performance enough for edge-centric rotating machinery fault diagnosis. However, this device has the next limitations that should be considered during application development. First, the built-in static random access memory (SRAM) is limited down to 8 MB, two of which are dedicated to the CNN accelerator. This means that neural network runtime data must not exceed 2 MB and executable code of control algorithm along with neural network weights should not exceed 6 MB. For sequential CNN, the runtime data could be estimated as a maximum sum of feature maps of two sequential layers. Second, as this device designed to be low-power and mobile, its performance is restricted, and standard CPU frequency is reduced down to 400 MHz. To reach maximum performance, most of the operations should be expressed through convolution to be processed by the CNN hardware accelerator.

Below we describe CNN developed to perform rotating machinery fault diagnosis running on the edge neural network computing device.

B. CNN for vibration signal based bearings fault detection

In this paper, we use the CNN for vibration signal based bearings fault detection. The input data of purposed CNN is raw vibration signal from the accelerometer installed close to the bearing. The CNN is composed of the sequence of three 1-D convolutional and pooling layers, followed by two fully-connected layers. The last layer consists of three nodes, corresponding to three detected classes: healthy bearing, damage of the outer ring, and damage of the inner ring. The concept of application of CNN to vibration signal classification, as a special case of time series classification, consists of follows.

The first convolutional layer applies multiple filters to the input 1-D tensor. Each filter has an individual convolution kernel for each channel of the input tensor. As the input of the first layer is only a vibration signal, input tensor has only one channel, so each filter has one kernel and applies one convolution to the input tensor. As there are multiple filters in the layer, this produces multiple outputs. In the case of a 2-D convolutional layer applied to spatial data (such as an image), the outputs are also is two-dimensional and represent spatial feature distribution, so the set of these outputs is referred to as a feature map, where each channel corresponds to the certain feature. In the case of raw vibration signal processing, we convolve signal (1-D vector) along the time axis by 1-D convolutions and obtain a set of 1-D vectors represented time distribution of features. We will refer to this set as a feature map and distinguish individual 1-D vector as a channel, to preserve common terminology.

Each convolution could be treated as filtering, or as a cross-correlation between raw signal and certain pattern. The form of filter kernel or cross-correlation pattern is defined during the training process. This allows CNN to automatically learn and extract features that best describe the data.

After the convolutional layer, the pooling layer is applied.

 ISBN: 978-1-61208-811-2

It performs down-sampling of the feature map. We use max-pooling for all layers. It allows us to preserve the most important information and significantly reduce feature map size and computation amount.

The next two couple of convolutional and max-pooling layers performs extraction features of an ever-higher level of abstraction. Finally, the last two fully-connected layers perform classification based on the extracted features.

The hyperparameters of CNN, such as the number of layers, number of units in fully-connected layers, number of filters and size of kernels in convolutional layers, have been selected through manual search and tuning with the aim to maximize accuracy on the validation set and prevent the overfitting.

The input of CNN is the 1-D tensor of 8192 samples of normalized vibration signal recorded at 64 kHz sample rate, which equals to 128 ms. Considering that the lowest rotation speed is 900 rpm, the full revolution takes approximately 67 ms or less. Thus, 128 ms should be sufficient to detect distinctive vibration produced by defects. The output of CNN is the 1-D tensor of three elements corresponded to the probabilities of detected classes. To train CNN we use Adam optimizer [12] with learning rate 0.00001 and categorical cross-entropy loss function. The number of training epoch is determined during the training process by monitoring the validation loss and when it has stopped decreasing the training process is terminated. The detailed description of used CNN is shown in Table III.

TABLE III. DESCRIPTION OF CNN HYPERPARAMETERS.

Layer	*Shape*	*Parameters*	
Input	(8192)		
1D Convolutional Layer	(8192, 2)	Activation	ReLU
		Filters	2
		Kernel Size	64
		Stride	1
		Padding	Same
1D Pooling Layer	(512, 2)	Pool size	16
1D Convolutional Layer	(512, 12)	Activation	ReLU
		Filters	12
		Kernel Size	32
		Stride	1
		Padding	Same
1D Pooling Layer	(32, 12)	Pool size	16
1D Convolutional Layer	(32, 32)	Activation	ReLU
		Filters	32
		Kernel Size	16
		Stride	1
		Padding	Same
1D Pooling Layer	(2, 32)	Pool size	16
Fully-connected layer	(150)	Activation	Sigmoid
		Units	150
Fully-connected layer	(3)	Activation	Softmax
		Units	3

For the training and evaluation of CNN, we use Paderborn University Dataset [3]. We select five bearings for each class. For outer ring damage and inner ring damage classes, the bearings with real damages have been chosen. The categorization of the dataset is given in Table IV. On the basis that, in the practical application, the monitored physical objects (bearings) differ from ones used in the training process, the validation split has been made by bearing's name rather than random examples splitting. We split the dataset into two parts: training set (set 2-5 in Table IV) and validation set (set 1 in Table IV). As recorded signal length in each file in the used dataset is 4 seconds, but CNN input length is 128 ms, at each training step a random frame of 128 ms is selected from random file from the dataset.

TABLE IV. CATEGORIZATION OF DATASET.

Set No.	*Healthy (Class 1)*	*Outer ring damage (Class 2)*	*Inner ring damage (Class 3)*
1	K001	KA04	KI04
2	K002	KA15	KI14
3	K003	KA16	KI16
4	K004	KA22	KI18
5	K005	KA30	KI21

To implement CNN, we use Keras with TensorFlow backend. To deploy CNN to Kendryte K210 system-on-chip, we convert our CNN to the TensorFlow Lite FlatBuffer file (.tflite) [13], and next, we compile it by nncase utility [14] to KModel format, which could be executed on the Kendryte K210 with hardware acceleration of convolution.

V. FEASIBILITY STUDY

In this section, to test the applicability of proposed CNN for bearings fault detection and classification, we train and test on Paderborn University Bearing Dataset [3]. We analyze obtained learning curves and confusion matrices. To test the portability of proposed CNN to edge devices, we deploy trained CNN to Kendryte K210 system-on-chip and evaluate its performance.

A. CNN training and testing

The set of training trials with fixed hyperparameters and random weights initialization has been conducted and the best one had been selected. The learning curves of CNN accuracy and loss on the training and validation datasets are shown in Figures 2 and 3. The result shows that after approximately 60 training epochs the validation loss is steady at the same level, so the training process had been terminated. The validation accuracy steady at 88%.

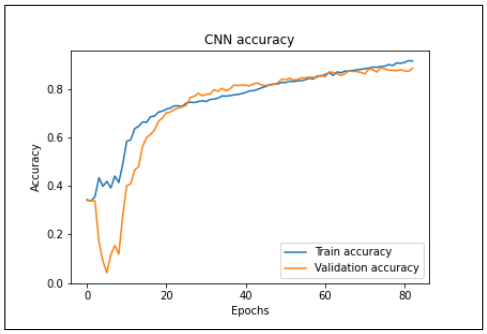

Figure 2. Training and validation accuracy curves of CNN.

The confusion matrices on training and validation data are shown in Figures 4 and 5, respectively. It could be noticed that obtained CNN shows the high classification accuracy both on training and validation data. This result allows us to use CNN not only for fault detection tasks, but also for determine the fault type.

B. CNN performance evaluation on the Edge Device

After CNN had been trained used Keras framework with TensorFlow backend, it had been deployed to Kendryte K210 system-on-chip. Obtained CNN requires 23 660 bytes to store its structure and weights as well as 81 920 bytes to store intermediate features maps and other runtime data during the forward pass. Consequently, the method is lightweight enough to run on the edge device like Kendryte K210.

 ISBN: 978-1-61208-811-2

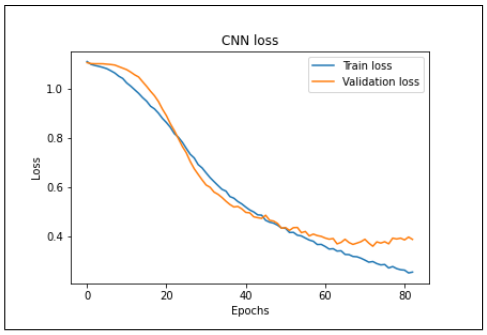

Figure 3. Training and validation loss curves of CNN.

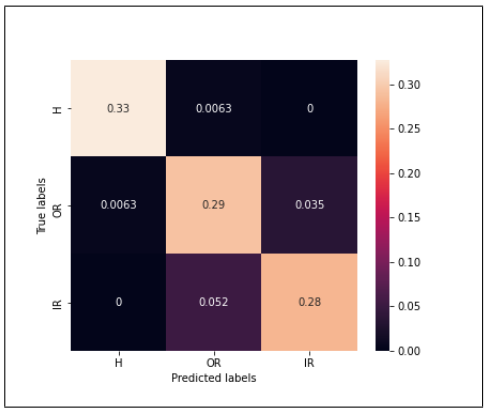

Figure 4. Confusion matrix of CNN on training data normalized over all population.

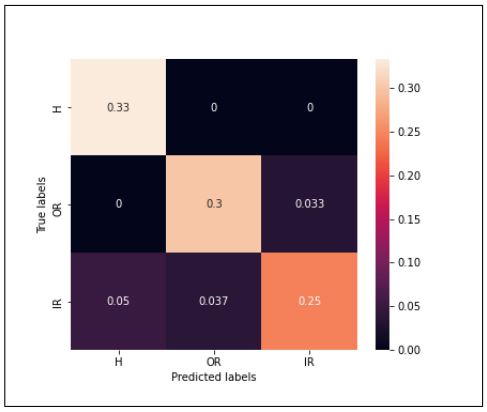

Figure 5. Confusion matrix of CNN on validation data normalized over all population.

The measured neural network execution time was 212 ms, which is approximately 1.66 times larger than the used frame size of the signal. Since in real applications, depending on the monitoring object, it is sufficient to carry out diagnostics once every 1…10 seconds, the purposed system could be able to monitor up to 50 units in real-time. In case the units are different and different CNN have to be used, it is possible to store in SRAM more than one CNN and use them in turn.

Depending on memory consumption by other applications, more than 200 CNNs like the one used in this paper could store simultaneously in SRAM. However, in this case, additional time would be spent on resources initialization and releasing. In our experiments, the initialization time of CNN and resources releasing time are about 3 ms and 90 us, respectively, which are significantly less than the CNN execution time.

The detailed analysis of the computation graph of CNN ported to Kendryte K210 revealed that only matrix multiplication in the first fully-connected layer is accelerated by the CNN hardware accelerator. This acceleration is possible by replacing of matrix multiplication by multiple convolutions, which is performed by neural network compiler nncase. The replacing consist in the conversion of matrix product of the weight matrix of shape (N, K) by the activation matrix of shape $(K, 1)$ to convolutional layer, whose number of filters is equal N and input is activation matrix of shape $(K, 1, 1)$, where the first dimension is channel number.

However, convolutional layers are processed by CPU without hardware acceleration, while those layers include most of the calculations. This is because CNN accelerator unit is only able to perform 1×1 or 3×3 convolution, while convolutional layers of our CNN have 1×64, 1×32, and 1×16 kernels, and there is no an algorithm in nncase utility to transform arbitrary convolution into a set of 1×1 or 3×3 convolutions. Hence, to further improve the performance, we need to develop CNN consist only of 1×1 or 3×3 convolutions or develop a method of transformation of arbitrary convolution into a set of 1×1 or 3×3 convolutions.

VI. Conclusion

This paper discussed the edge computing opportunities for fault diagnostics in industrial rotary machinery using CNN methods. We showed that the edge IoT device capacity is enough to make data analysis based on CNN. The analysis can be implemented in real-time, so online data analytics services can be provided to personnel near or remote the industrial production system.

Acknowledgment

This research is implemented in Petrozavodsk State University (PetrSU) with financial support by the Ministry of Science and Higher Education of Russia within Agreement no. 075-11-2019-088 of 20.12.2019 on the topic “Creating the high-tech production of mobile microprocessor computing modules based on SiP and PoP technology for smart data collection, mining, and interaction with surrounding sources”.

References

[1] D. Hasselquist, A. Rawat, and A. Gurtov, “Trends and detection avoidance of internet-connected industrial control systems,” IEEE Access, vol. 7, 2019, pp. 155 504–155 512.

[2] K. Zhong, M. Han, and B. Han, “Data-driven based fault prognosis for industrial systems: a concise overview,” IEEE/CAA Journal of Automatica Sinica, vol. 7, no. 2, 2020, pp. 330–345.

[3] C. Lessmeier, J. K. Kimotho, D. Zimmer, and W. Sextro, “Condition monitoring of bearing damage in electromechanical drive systems by using motor current signals of electric motors: A benchmark data set for data-driven classification,” in Proceedings of the European conference of the prognostics and health management society, 2016, pp. 05–08.

[4] T. A. Shifat and J. W. Hur, “An effective stator fault diagnosis framework of bldc motor based on vibration and current signals,” IEEE Access, vol. 8, 2020, pp. 106 968–106 981.

[5] S. Naveen and M. R. Kounte, “Key technologies and challenges in iot edge computing,” in 2019 Third International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), 2019, pp. 61–65.

 ISBN: 978-1-61208-811-2

[6] L. Jing, T. Wang, M. Zhao, and P. Wang, “An adaptive multi-sensor data fusion method based on deep convolutional neural networks for fault diagnosis of planetary gearbox,” Sensors, vol. 17, 02 2017, pp. 1–15.

[7] J.-R. Jiang, J.-E. Lee, and Y.-M. Zeng, “Time series multiple channel convolutional neural network with attention-based long short-term memory for predicting bearing remaining useful life,” Sensors, vol. 20, no. 1, 2020, pp. 1–19.

[8] J. K. Kimotho and W. Sextro, “An approach for feature extraction and selection from non-trending data for machinery prognosis,” in Proceedings of the second european conference of the prognostics and health management society, vol. 5, no. 4, 2014, pp. 1–8.

[9] D. Palossi et al., “A 64-mw dnn-based visual navigation engine for autonomous nano-drones,” IEEE Internet of Things Journal, vol. 6, no. 5, 2019, pp. 8357–8371.

[10] D. Park, S. Kim, Y. An, and J.-Y. Jung, “Lired: A light-weight real-time fault detection system for edge computing using lstm recurrent neural networks,” Sensors, vol. 18, no. 7, 2018, p. 2110.

[11] “K210 Datasheet,” 2019, URL: https://s3.cn-north-1.amazonaws.com.cn/dl.kendryte.com/documents/kendryte_datasheet_20181011163248_en.pdf [accessed: 2020-08-26].

[12] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” arXiv preprint arXiv:1412.6980, 2014, pp. 1–15.

[13] “TensorFlow Lite converter,” 2020, URL: https://www.tensorflow.org/lite/convert [accessed: 2020-08-26].

[14] “GitHub - kendryte/nncase: Open deep learning compiler stack for Kendryte K210 AI accelerator,” 2020, URL: https://github.com/kendryte/nncase [accessed: 2020-08-26].

 ISBN: 978-1-61208-811-2