Anomaly Detection Using Edge Computing AI on Low Powered Devices

Dragoș-Vasile Bratu, Rareş Ilinoiu, Alexandru Cristea, Maria-Alexandra Zolya, Sorin-Aurel Moraru

Anomaly detection using Edge Computing AI on Low Powered Devices

Dragoș-Vasile Bratu[1], Rareș Ștefan Tiberius Ilinoiu[2], Alexandru Cristea[3], Maria-Alexandra Zolya[4], and Sorin-Aurel Moraru[5]

[1] Transilvania University of Brașov, Brașov 500036, Romania
d.v.bratu@gmail.com
[2] Transilvania University of Brașov, Brașov 500036, Romania
rares.stefan.ilinoiu12@gmail.com
[3] Transilvania University of Brașov, Brașov 500036, Romania
alex.v.cristea@gmail.com
[4] Transilvania University of Brașov, Brașov 500036, Romania
alexandrazolya18@gmail.com
[5] Transilvania University of Brașov, Brașov 500036, Romania
smoraru@unitbv.ro

Abstract. In the industrial environment, maintaining a permanent good state of functioning for every piece of equipment has a substantial importance. This, however, is very difficult to attain, due to the mechanical wear, the environment of operation, or improper usage. Predictive maintenance is a practice that is performed to determine the condition of the machinery in service and estimate the time when the maintenance should occur. The challenge of detecting a possible fault in a piece of equipment before it occurs is one of the main tasks of the predictive maintenance process. Reading data from sensors and creating firmware that monitors the equipment can be time and resource-consuming, and not practical if the equipment is changed frequently. Nowadays, the computational power of Artificial Intelligence exceeds that of a computer. As the industrial equipment and the hardware components of a conventional computer are getting increasingly expensive and demanded, more and more entities are running Machine Learning algorithms, which make the data exchange with a server that runs this service a more feasible process. This approach poses several challenges due to latency, privacy, bandwidth, and network connectivity. To solve these limitations, computation should be moved as much as possible towards the Edge, directly on the devices that gather the data. In this article, we propose a compact and low-powered solution that is accurate and small enough to be fitted on a microcontroller or a device that runs on the Edge. This approach ensures that a minimum amount of resources are used. The solution consists of an Unsupervised learning algorithm that can detect anomalies in the vibration patterns of the bearings or the casing of industrial motors. It uses an Autoencoder that takes as input the median absolute deviation of each measurement set provided by an accelerometer, then with the help of a classifier compares the values provided by the output to values that are known to be normal vibration patterns and decides if it deals with an anomaly or not. The low-powered Edge device is an ESP32

board that consumes only 160mAh on full load but also being powerful enough to maintain WiFi and Bluetooth capabilities when needed. On a more economical operating mode, without WiFi and Bluetooth capabilities it can consume as low as 3mAh [1]. This feature and the fact that the board is connected directly to the data-gathering sensor makes it preferable to an algorithm hosted on a remote server or a local machine due to low resource consumption and easy maintainability. The Autoencoder is fitted on this board and runs continuously until it encounters an anomaly, which in turn provokes an alert to the user.

Keywords: ESP32 · Autoencoder · Machine Learning · Edge Computing · Anomaly Detection

1 Introduction

In a study performed by Eurostat [2] it was discovered that 41% of EU enterprises used Cloud Computing in 2021 and 73% of those enterprises used sophisticated Cloud services relating to security software applications, hosting enterprise's databases, or computing platforms for application development, testing or deployment. Compared with 2020, the use of Cloud Computing increased by 5 percentage points. In this paper, a step forward in this direction was made, by proposing a cost-effective IoT solution running on Edge that can perform anomaly detection right on the data-generating device. Generally speaking, anomaly detection is a branch of Artificial Intelligence (AI) that deals with the identification of patterns in a set of data that do not correspond to a normal behavior definition agreed upon beforehand. To detect anomalies, a model that describes the patterns of a normal behavior has to be defined first. Only then, the anomalies can be successfully detected by the model.

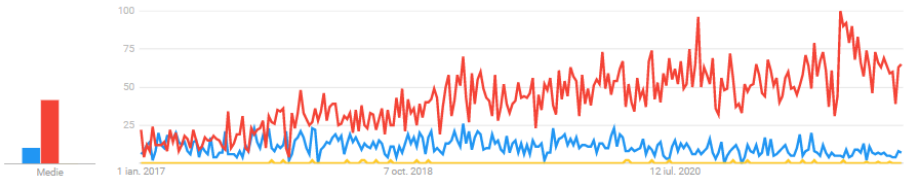

Fig. 1. Comparison between trends of Edge, Fog and Mist computing.

2 Related work

Anomaly detection in gas turbines described in [8] is achieved with the help of a Deep Autoencoder (DAE) which analyzes the performance state of the turbine, more exactly including DEGT, Exhaust Gas Temperature Margin (EGTM), Delta Fuel Flow (DFF), Delta Core Speed(DN2). Their solution utilizes two traditional DAEs and a k-means clustering model. One DAE together with the

k-means clustering model is used for a sample selection mechanism to build the original training set, while another DAE is used to calculate the reconstruction error and high-level feature of each original sample [8]. With the help of the bearing data set an anomaly detection solution using a CNN and GRU was developed as described in [9]. The latter article suggests making use of the temporal component of the data with a novel deep architecture named stacked CNN and GRU combination network (SCG network) to predict the anomaly values. The significant difference between the neural network architecture in this paper and those cited above (esp. those in [7], [6]) is the fact that the whole system is built in a robust way, specifically to be fitted into a microcontroller that runs on the Edge with an Autoencoder.

2.1 System model and overview of approach

The goal of this project was to develop a Machine Learning algorithm, which can detect abnormalities in the vibration patterns of an AC motor, preventing in this way failures of the equipment and reducing the time necessary for maintenance. Such an algorithm is very important because it supports a predictive [10] approach to device maintenance [11], thus preventing the user or the person responsible for managing the equipment from the undesirable situation of finding that the equipment has sustained severe damage.

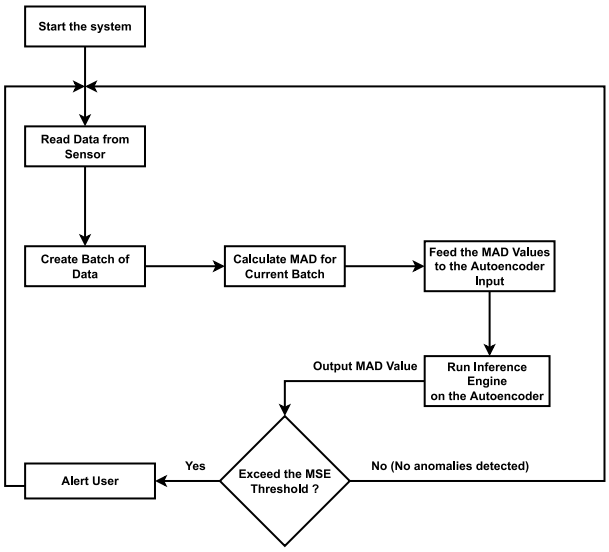

Fig. 2. Workflow diagram of the system

With the support of this method, the user or the person in charge of the equipment will be able to be notified in advance of a possible decay of the health of the motor, and in turn, take actions to prevent such an outcome. The vibration data coming from electric motor bearings was analyzed. Bearings are

a key factor in deciding if the system is prone to faulty behavior. Since bearings are moving parts, they are more likely to break down due to heat dilatation, dust particles and wear over time. All these factors can influence the lifespan of the bearings. The Neural Network that was developed in this scope has the task of analyzing the vibration data as it's being produced and determining whether the motor is likely to break in the future. The model was trained with data until the predictions it makes are accurate enough for deployment on the production line. An algorithm for predicting and detecting anomalies can be very demanding from a computational point of view [12]. Making sure that the model is fast, small and reliable enough to run on a microcontroller is crucial.

After the training phase, the model is being compressed until it is small enough to fit on a chip. The goal was to deploy the model on an IoT Edge device that runs alongside the data-producing system.

3 Model Development

As discussed above, the first phase was designing a model that is fitting to the problem of vibration analysis. Vibration data and anomaly detection problems often imply that the data is unlabeled. Another point worth mentioning is that raw data, especially when it comes to vibration, needs to be filtered as, more often than not, it is altered by noise. Simple low-pass filtering is usually enough for most applications, but in order to make an accurate and reliable model, it is necessary to make sure that the network only deals with clean data. To achieve this, an Autoencoder architecture was used, as seen in Figure 3, for the network, which by construction, also filters out the noise from the data[13],[14].

The model was designed to consist of 5 layers, 3 of which are hidden. It also features a regularization layer that prevents the problem of overfitting. Placed after the first hidden layer it features a rate of 0.2 and a (None, 3) shape. The design of this network is relatively simple, the number of inputs being equivalent to the number of outputs. The model has an input size of (8, None) as there are 4 bearings measured on 2 channels each. In contrast, in the hidden layers, the number of neurons decreases considerably to extract only the essential information from the training data, which allows the noise to be eliminated from the information that is being processed. The first hidden layer has an input size of (5, None), whereas the second (middle) has only 3 neurons with an input size of (3, None). This process is called encoding and decoding. The information that is fed into the network is compressed (encoded) inside the hidden layers and then decoded (decompressed) in the output layer. The fidelity with which the output data is reproduced shows the efficiency of the model. In order to determine the number of neurons and layers that would provide reasonable accuracy, we had an experimental approach aimed at training and testing several network architectures and keeping the one that provided the best results. From the repeated reconstruction of the architecture, we concluded that the activation function suited for this project is the rectified linear activation function also known as

ReLu. This provided the best results in terms of accuracy, as opposed to a linear one.

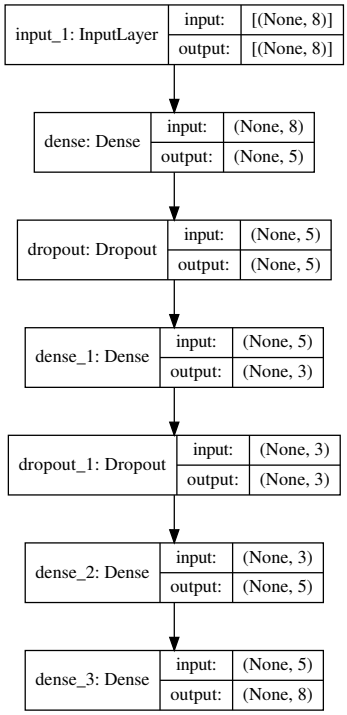

Fig. 3. Model architecture (generated with with Graphwiz and pydot)

3.1 Data preliminary processing

In order to train the network, a suitable data set is required. The set has to correspond to contain data coming from real devices that are exposed to vibration stress and are prone to malfunction because of it. The "NASA Bearing Data set" was chosen because it contains data recorded from industrial motor bearings that were run until failure. According to [15] four bearings that were installed on a shaft were subjected to a radial load of 6000lbs (approx. 2721.554Kg) applied to the shaft and bearing while kept on a constant rotation speed of 2000RPM. Each data set consists of individual files that are 1-second vibration signal snapshots recorded at specific intervals. Each file consists of 20,480 points with the sampling rate set at 20 kHz. The development of a reliable model requires, in the beginning, a consistent amount of data. The samples that are being used for the training data set should contain both a normal operating behaviour as well as a visible anomaly. As seen in Figure 4, this particular data set satisfies the above-mentioned requirements.

The next step was making the feature extraction and the outlier identification. Although sometimes raw data is enough for the training of a model, usually

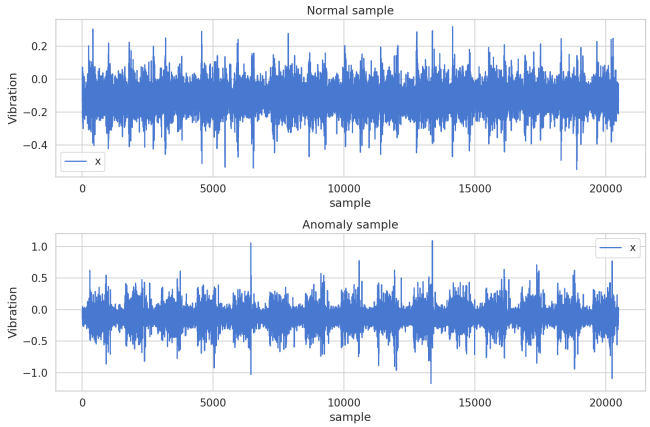

Fig. 4. Comparison between a normal vibration sample and an anomaly one

a necessary part of the process is to determine specific features that the Neural Network is going to analyze. This has great implications for the algorithm speed, due to the fact that analyzing raw data is always going to be more computationally expensive than analyzing a filtered version of it. To achieve this, the DC component of the signal was removed by subtracting the mean amplitude. This gives the possibility to visualize the vibration component of the signal and visually identify the outliers as seen in Figure 5.

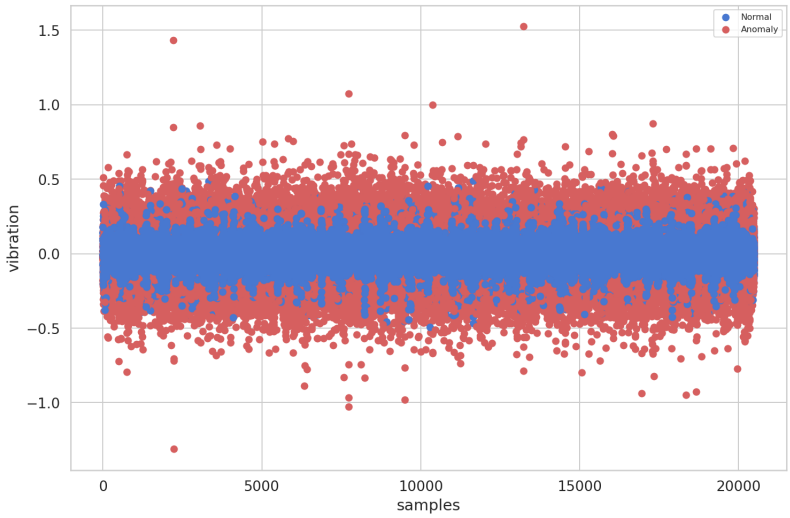

Fig. 5. Healthy and Abnormal vibration data points with DC component removed

It is easy to notice how considerably the anomaly sample in red differs from the normal one colored in blue. The red one is much more spread, whereas the

blue one is more compact. Further analysis of the data reveals an even more evident distinction between the two. Further, the Variance, Skew, Kurtosis, and Median Absolute Deviation (MAD) were analyzed. Considering the variance in Figure 8, average squared differences were measured from the Mean or how far each value from the data set was from the mean. In this way, the data is linearly separable.

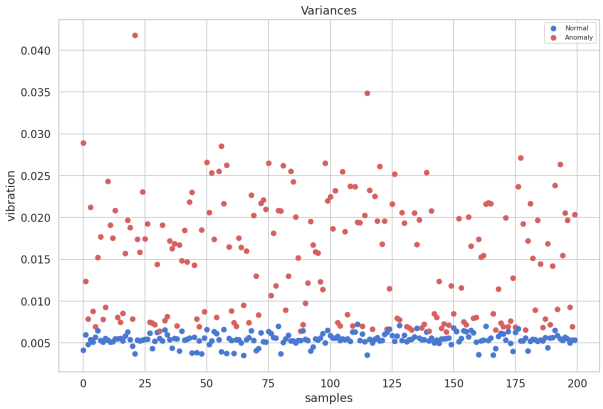

Fig. 6. Comparison between variance of a normal and anomalous vibration samples

High Kurtosis in a data set is an indicator of the existence of outliers and it is used to describe the extreme values relative to a Gaussian distribution. Figure 7 illustrates the above. The mean absolute deviation of a data set is the average distance between each data point and the mean. This confirms the variability in the data set.

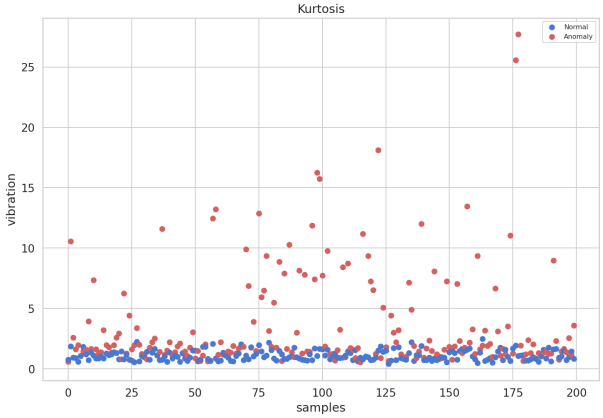

Fig. 7. Comparison between kurtosis of a normal and anomalous vibration samples

While standard deviation (and variance) are exceptional at describing the spread of data in a normal distribution, they can easily be affected by outliers

and non-normally distributed data. As a result, MAD offers a more robust way [16] to measure spread for non-normal data as seen in Figure 8.

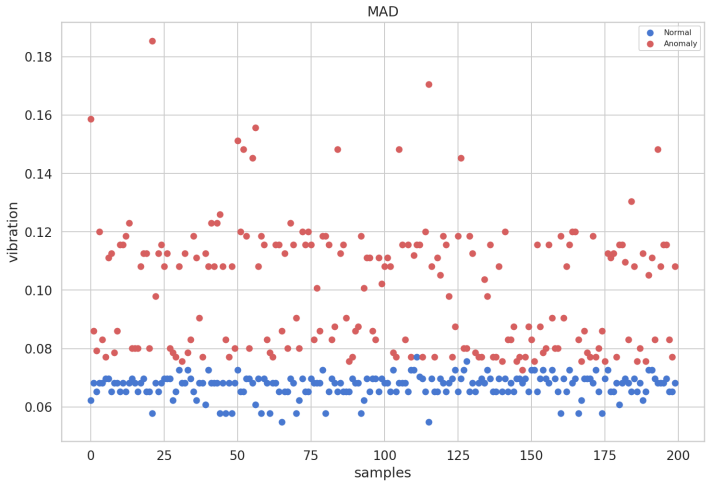

Fig. 8. Comparison between MAD component of a normal and anomalous vibration sample

As can be observed there is a more linear separability between the data that presents an anomaly in the MAD graphic plot. In this way, the algorithm will identify faster and easier the discrepancies between the two types of data.

3.2 Model training

For training the model, data were split into training (∼55%), cross-validation (∼10%), and testing sets (∼35%). The training set is characterized by the fact that it contains non-anomalous data exclusively. This is because the Autoencoder should train to recognize a normal operation of a motor very well so that everything that does not resemble a normal behavior will be classified as an abnormality.

In our case, the data prior to the bearing failure in the motor is used as a training set. For the validation set, both normal and anomalous data were used. At this step, the network is expected to be able to recognize with good accuracy both normal and anomalous data. The test set consists of random data, similar to what the network would receive from the motors in real-time. Training is done for 50 epochs with a batch size of 55, the optimizer used for this matter being 'Adam' and the loss function being the mean squared error or "MSE". Weights are generated randomly at the beginning of the training process.

Plotting the loss function graph during training for both test and validation sets reveals that the model performed overall with good results, reaching convergence as seen in Figure 9.

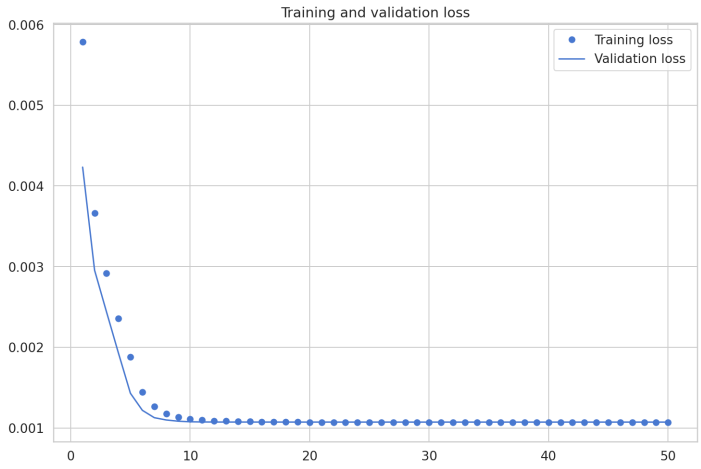

Fig. 9. Comparison between training and validation set learning curve

Plotting the histograms of normal versus anomaly training sets, in figure 10, reveals that there is a clear separation between the Mean Squared Error(MSE) of the loss function.

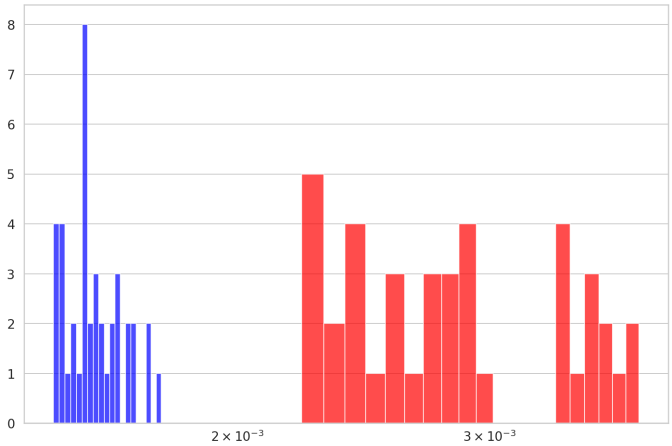

Fig. 10. Comparison between training and validation set Mean Squared Errors

4 Testing the solution

At this step, constructing a classifier is fairly easy. The last mean squared error value from the validation set was chosen as a threshold. Everything above this will be considered an anomaly. With the help of a confusion matrix, the performance indices of the model can be visualized as can be noticed in Figure 11. The solution was saved with a .tflite extension(as TensorFlow Lite was used)

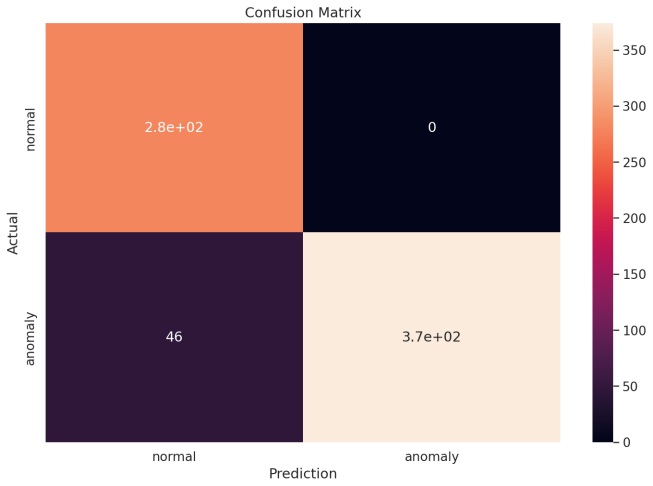

Fig. 11. Confusion matrix (on validation set)

in order to be flashed on a microcontroller and more concrete on the ESP32 microcontroller.

Table 1. Comparison between testing with and without inference.

Phase/Method	No. of samples	No. of samples in each file	Computing Time (μs)	Contains anomalies	Error (%)
Testing(only MAD)	200	20 000	2102	no	6.57
Testing(only MAD)	200	20 000	2153	yes	6.57
Testing (MSE + Inference	200	20 000	150	no	6.57
Testing (MSE + Inference)	200	20 000	45	yes	6.57

In the testing phase, it was observed that the model flashed on the ESP32 microcontroller is performing remarkably well (∼93.42%) in detecting anomalous versus normal data, on data points not seen before by the system. For a normal test sample, that was provided for calibration purposes, the algorithm took 6795μs to compute the MAD. The inference result for predicting unseen before data was completed in 161μs. The time for running inference and computing the MAD for an anomaly sample was 106μs. By comparing the results of what was run in the test environment, in the model design phase, and what was obtained after running the algorithm on target, it was discovered that the error between the predicted values is less than 0.1. This indicates that the algorithm is secure and reliable for deploying and classifying the data in a good manner.

5 Conclusion

Extrapolating on the basis of the article presented, the project has reached a safe phase of operation on the side of anomaly detection and no major problems were found in any of the tests performed. The algorithm performed remarkably well (∼93.42%) in the testing phase. The algorithm is fully developed as described above in the project and is planned to be put into operation on the production line as soon as possible. The Autoencoder can then take the previously computed MAD values as input and try to recreate it as fairly in the output. The classifier then takes the output MAD values of the autoencoder and compares them to a threshold which gives us a signal if we are indeed dealing with an anomalous sample or not. Given that the article addresses a cutting-edge topic in Machine Learning, it may be subject to major transformation and further development in the future from both the platform used (Tensorflow Lite) but also in the overall structure.

The novelty of Edge Computing gives this project a great versatility, the idea being applied in several fields, and the fact that this niche is constantly developing allows updating and optimizing the solution with the latest methods and technologies in the field.

References

1. B. Suwandi, T. Kitasuka, M. Aritsugi, "Vehicle Vibration Error Compensation on IMU-accelerometer Sensor Using Adaptive Filter and Low-pass Filter Approaches" doi.org/10.2197/ipsjjip.27.33 https://www.jstage.jst.go.jp/article/ipsjjip/27/0/27_33/_article/-char/ja/
2. Cloud computing - statistics on the use by enterprises https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Cloud_computing_-_statistics_on_the_use_by_enterprises#Use_of_cloud_computing:_highlights
3. M. Fahim and A. Sillitti, "Anomaly Detection, Analysis and Prediction Techniques in IoT Environment: A Systematic Literature Review," in IEEE Access, vol. 7, pp. 81664-81681, 2019, doi: 10.1109/ACCESS.2019.2921912
4. Huang, Huiyue, et al. "Digital twin-driven online anomaly detection for an automation system based on edge intelligence." Journal of Manufacturing Systems 59 (2021): 138-150
5. Finke, T., Krämer, M., Morandini, A. et al. "Autoencoders for unsupervised anomaly detection in high energy physics" J. High Energ. Phys. 2021, 161 (2021). https://doi.org/10.1007/JHEP06(2021)161
6. Gohel, Hardik A., et al. "Predictive maintenance architecture development for nuclear infrastructure using machine learning." Nuclear Engineering and Technology 52.7 (2020): 1436-1442
7. M. Antonini, M. Vecchio, F. Antonelli, P. Ducange and C. Perera, "Smart Audio Sensors in the Internet of Things Edge for Anomaly Detection," in IEEE Access, vol. 6, pp. 67594-67610, 2018, doi: 10.1109/ACCESS.2018.2877523
8. S. Fu, S. Zhong, L. Lin, M. Zhao, "A re-optimized deep auto-encoder for gas turbine unsupervised anomaly detection" Engineering Applications of Artificial Intelligence, Volume 101, 2021,

104199, ISSN 0952-1976, https://doi.org/10.1016/j.engappai.2021.104199. (https://www.sciencedirect.com/science/article/pii/S0952197621000464)
9. K. Lee, J. -K. Kim, J. Kim, K. Hur and H. Kim, "CNN and GRU combination scheme for Bearing Anomaly Detection in Rotating Machinery Health Monitoring," 2018 1st IEEE International Conference on Knowledge Innovation and Invention (ICKII), 2018, pp. 102-105, doi: 10.1109/ICKII.2018.8569155
10. Mobley, R. K. (2002). Book "AN INTRODUCTION TO PREDICTIVE MAINTENANCE"
11. Xh. Mehmeti, B. Mehmeti, Rr. Sejdiu, "The equipment maintenance management in manufacturing enterprises" ,IFAC-PapersOnLine,Volume 51, Issue 30,2018,Pages 800-802,ISSN 2405-8963,https://doi.org/10.1016/j.ifacol.2018.11.192.
12. B. Burton and R. G. Harley, "Reducing the computational demands of continually online-trained artificial neural networks for system identification and control of fast processes," in IEEE Transactions on Industry Applications, vol. 34, no. 3, pp. 589-596, May-June 1998, doi: 10.1109/28.673730
13. "Deep denoising autoencoder for seismic random noise attenuation" Omar M. Saad, Yangkang Chen; Deep denoising autoencoder for seismic random noise attenuation. Geophysics 2020;; 85 (4): V367–V376. doi: https://doi.org/10.1190/geo2019-0468.1
14. N. M. N. Leite, E. T. Pereira, E. C. Gurjão and L. R. Veloso, "Deep Convolutional Autoencoder for EEG Noise Filtering," 2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), 2018, pp. 2605-2612, doi: 10.1109/BIBM.2018.8621080
15. J. Lee, H. Qiu, G. Yu, J. Lin, and Rexnord Technical Services (2007). IMS, University of Cincinnati. "Bearing Data Set", NASA Ames Prognostics Data Repository (http://ti.arc.nasa.gov/project/prognostic-data-repository), NASA Ames Research Center, Moffett Field, CA
16. C. Leys, O. Klein, P. Bernard, L. Licata, "Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median", Journal of Experimental Social Psychology, Volume 49, Issue 4, 2013, Pages 764-766, ISSN 0022-1031, https://doi.org/10.1016/j.jesp.2013.03.013. (https://www.sciencedirect.com/science/article/pii/S0022103113000668)