Journal of Ubiquitous Systems & Pervasive Networks
Volume 3, No. 1 (2011) pp. 00-00

Use Cases of Data Reduction to Time Series Data in Sensor Monitoring

Selvine G. Mathias [a]*, Daniel Grossmann [a]

[a] *Technische Hochschule Ingolstadt, Esplanade 10, 85049 Ingolstadt, Germany*

Abstract

The aim of this paper is to investigate the use of data reduction techniques using distances and areas for monitoring of sensors using process mining approaches. When sensors are used in industrial cases, the real time accumulation of timestamped data tends to pose a problem of storage, processing and analyzing their values in the optimal way possible. Here, the paper tends to present an application of monitoring of sensor signals using reduced parameters from the acquired timestamped values. Each observation is dissected into packets and their attributes such as areas under the curves and successive distances are calculated. The combination of these attributes present real time monitoring scenario for which a blueprint process model can be constructed. This, in turn, helps in identifying signal variations during run-time of the sensor without advanced analysis.

Keywords: *data reduction, sensor, use case, process model, etc.*

1. Introduction

Huge volume of datasets in different fields leaves the task of determining the optimal way to clean, pre-process and analyze it, on the user, depending on the gravity of problem at hand. Especially in the cases of special data, such as time series, where each sample is dependent on the previous observation, the reduction of data volume is a scientific challenge. Heterogeneous values from different sensors and devices present real world investigations using tasks such as pattern and behavioral changes detection over time, life expectancy of products, fault prognosis using assessment, analysis and predictions. These problems need additional mechanisms such as data processing and Machine Learning (ML) for decision-making tasks. Present approaches of data reduction using known techniques such as PCA or SVD Decomposition may or may not be optimal or feasible solutions. Using correlations and statistical analysis to a certain extent gives inference and reliability based metrics that help in deciding to cut down a number of features.

Time series data is a sequence of values that correspond to a interval based recording from devices such as sensors or events. Abundant examples are present in the form of weather parameters such as temperatures, financial values such as monetary values or economical indexes over time. The accumulation of sequential values over time demands some sort of either immediate visual analysis or a later in-depth analysis preceded by storage requirements. Either way, there needs to be a method to process this data in an easy manner rather than right away employ Big Data techniques requiring various expensive resources and platforms. Fortunately, with advances in data analysis, many different methods do exist to pre-process time-series data, though none are standardized that can be used uniformly across all kinds of time-series. It is in this context that we present reduction of data to a shortened representable version as an alternative that can be used instead of the original dataset. The focus of this paper is to be able to present simple monitoring applications that can be developed using the current programming languages after reducing and changing the data.

The paper is structured as follows: Section 2. presents the associated literature on works based on sensor data while Section 3. presents the proposed methodology. The implementations using a public dataset and outlines of two use cases based on the given scheme is also discussed here. The final section concludes the work of this paper with a brief outlook on future directions.

*Corresponding author. Tel.: +49-841-9348-6436
Fax: +49-841-9348-2000; E-mail: SelvineGeorge.Mathias@thi.de

DOI: 10.5383/JUSPN.03.01.000

2. Similar Works in Data Reductions

In data reduction related problems, an extensive literature is available. Some well known feature reduction techniques employed are Principal Component Analysis (PCA) and feature selections or extractions using ensemble classifiers. A brief summary of available techniques along with the application of a new agent-based population algorithm using rotation, stacking and classification algorithms in comparison to these are presented in [1]. However, these experiments were performed on public datasets which were not temporal.

In [2], the authors discuss condensing a large dataset into a small set of informative samples for training deep neural networks from scratch. The formulation involves using gradient matching problem between the gradients of a deep neural network trained on the original data and the procured synthetic data. This way, a synthetic small dataset can be trained to be applied in bigger networks for continual learning, thereby effecting data training efficiency.

In [3], the authors propose a technique to process raw time-series data using deep learning networks. The conventional signal processing techniques for noise reduction and feature extraction (time and frequency domain) are not applied here. This minimizes the dependence of the machine learning model's performance on the quality of features extracted from the raw time-series data on one-hand and preserving the information of temporal coherence on time-series data on the other, which is usually lost due to feature extraction.

Generally speaking, current researches include building applications like computer-vision, bio-informatics, fraud-detection, Intrusion Detection Systems (IDS), face recognition, speech recognition, etc. based on mechanisms like ML and Big Data [4–6]. Most of the researches present applications based on raw sensor data such as [7] and [8]. The authors in former propose a detailed sensor data processing pipeline, which not only includes the mechanism of sensor data acquisition but also a fault prediction based on outlier detection. The mechanism uses Apache Kafka, Apache Storm and MongoDB as a message queue, real-time processing engine and a storage for sensor data from manufacturing process respectively, finally Density-Based Spatial Clustering of Applications with Noise (DBSCAN) and Random Forest Classification were used to detect outlier and predict fault. In the latter, the importance of predictive maintenance in machines has been discussed and emphasized using sensor data, which saves costs and prevents downtime. Here, the authors propose a system consisting of two sub-systems working in coherence to achieve the required goal of predictive maintenance - the sensing module and the analysis system. The sensing module consisted of sensor modules implemented on a programmable interface controller (PIC), while the information transmission was carried out using the TCP/IP protocol. The analysis system on the other hand uses a machine learning approach, i.e., the Decision Tree Model for prediction.

This paper extends the work in [9] to present implementable use cases in industries where data from sensors are accumulated over time, but due to non-standardization of analysis methods, the data remains non-usable. The two use cases are supplied with a business process model that formalizes this application and provides a possibility of enhancing product monitoring using sensors over time. The novelty of this paper lies in presenting the changed datasets as a representable and acceptable form of actual data, in applications as well as in analysis. The use cases demonstrate to a certain extent, that it is easier to work with a reduced version of sequential data in a different form, rather than considering the entire original dataset.

3. Data Reduction Methodology

To discuss the methodology, we begin with the description of the public dataset used. The reduction techniques follow with the implementations on the dataset and the use cases that can be extracted from this scheme.

3.1. Dataset Description

The data was generated by the NSF I/UCR Center for Intelligent Maintenance Systems (IMS – www.imscenter.net) with support from Rexnord Corp. in Milwaukee, WI [10]. Four bearings were installed on a shaft. The rotation speed was kept constant at 2000 RPM by an AC motor coupled to the shaft via rub belts. A radial load of 6000 lbs is applied onto the shaft and bearing by a spring mechanism and all bearings are force lubricated. Three data sets were collected based on three experiments. Two accelerometers were placed for each bearing [x- and y-axes] in dataset 1 while one accelerometer was placed for each bearing in datasets 2 and 3. All failures occurred after exceeding designed life time of the bearing which is more than 100 million revolutions. The setup is discussed briefly in [11].

Each dataset describes a test-to-failure experiment and it consists of individual files that are 1-second vibration signal snapshots recorded at 10 minutes interval where each file consists of 20,480 points with the sampling rate at 20 kHz. The file name indicates when the data was collected. Each row in the data file is a data point.

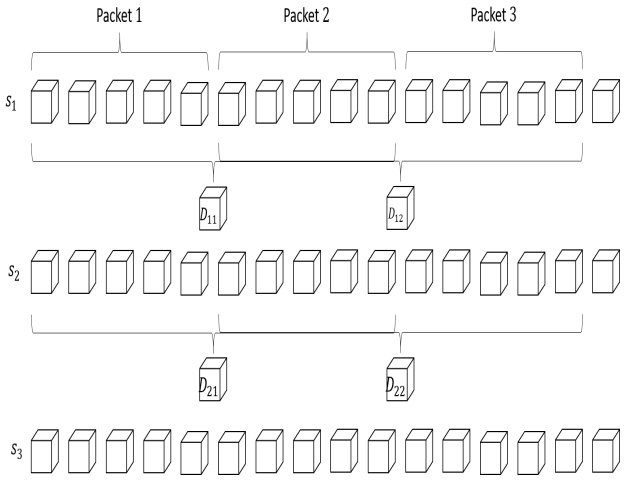

Fig. 1: Distance based reduction of samples [9]

3.2. Reduction and Parametrization Process

In industrial processes, huge batches of signals are produced at a timestamp from sensors. To reduce the data volume, either the frequency of the signal is modified beforehand or post-acquisition,

the features are reduced using statistical correlations, feature engineering or matrix decomposition. We propose another method as follows.
Each observed sample is divided into *packets* of equal length, where the number of packets is chose to cover as many points in the frequency range of the signal (see Fig. 1). Two parameters are derived from the reduced sample.

- Area under packet - this is calculated using trapezoidal rule
- Distance between packet - this is calculated using the euclidean distance formula. Other metrics can also be used in its stead.

We formalize this process here. A dataset D is a set $\{s_i\}$ of m samples with n features, where m, n can be very large. Let k be the length of a *packet* in a sample. So every s_i is divided in to $\frac{|s_i|}{k}$ packets. Since $|s_i| = n$, we have $\frac{n}{k}$ packets in every sample.
The Trapezoidal Rule for calculating area under a packet $s_{i,r}$ is given by

$$A_N(f) = \frac{\delta x}{2} \sum_{i=1}^{N} (f(x_i) + f(x_{i-1})) \tag{1}$$

where δx is the length of the sub-intervals, $f(x_i)$ is the observed value and N is the number of sub-intervals. For the purpose of simplicity, we set $\delta x = 1$ in this paper.
Equation (1) presents one rule of numerically integrating the area under an observed set of points. These formulas are readily available in many modern programming languages such as Python, R, MATLAB, etc.
If s_i is a sample, then the euclidean distance between two packets $s_{i,r}, s_{i,r+1}$ is given by

$$D_{ir} = \sqrt{(s_{i,r} - s_{i,r+1}) \circ (s_{i,r} - s_{i,r+1})} \tag{2}$$

Equation (2) presents the dot product formula of calculating euclidean distance between two equally sized vectors.
The consecutive distances between each of these $\frac{n}{k}$ packets form the reduced distance data for the same number of samples, while the average of areas between two consecutive packets form the reduced area data. In a time series, where variations from one extreme to another is obvious but not interpret-able without signal analysis, such reduced data can help in identifying increasing or decreasing trend. Although the actual samples are condensed in this version, we still have a time series which continuously gives the *closeness* of two consecutive packets using distance and areas. The original amplitude values of the vibration acceleration is however lost using this condensation. If similar trends can be extracted from the reduced data, then the original data may not be needed in some cases.
The foremost question is how to decide the length of packets. This is based on the knowledge of the data known prior to the application. For the above dataset, the packet length for reduction is obtained through the number of revolutions of the bearings per minute. Since each bearing is having a constant rotation speed at 2000 rpm, every second has about $\frac{2000}{60}$ rotations. Every file has 1-second snapshot so we can say that these $\frac{2000}{60}$ rotations are distributed in 20480 values. Therefore, we have about $\frac{20480*60}{2000} \approx 614.4$ values per rotation in a sample. To cover as many values as possible, a rotation is considered to be of length 620. This is the required packet length, so we have covered $620*33 = 20460$ values. Once every sample is distributed among consecutive packets, the areas under each packet and consecutive distances are calculated. For the purpose of uniformity, we calculate the average area between two consecutive packets. Thus, the reduced data of areas and distances comprise of 32 values among the 33 rotations in a sample.

3.3. Implementations and Discussions

We discuss two use cases for such reductions in practical applications as follows.

3.3.1. Use Case I: Visual On-Site Monitoring

Figure 2 shows the comparisons of actual packet values with the corresponding area and successive distances. The left most figure shows a visualization spanning the last 70 observed samples from the data, a total of $70 \times 20480 = 1433600$ points while the remaining two sub-figures correspond to the reductions of area and distance parameters comprising of $70 \times 32 = 2240$ values for each case of the data. The raw sensor values are too volatile to discern any noticeable changes or variations, but the average of two consecutive packet areas as well as the distance between them provide the first indicators that either the product concerned is showing defects or the sensor values are behaving unexpectedly. Either case warrants an immediate inspection. According to the presented visuals, we can categorize four indicators as follows:

- Non-Varying Area and Distance
- Gradual Increase in Area and Distance
- Early Variation in Area or Distance
- Sudden Variations in both Area and Distance

At this stage of investigation, we do not focus on the assignment of the a particular indicator to an observed defect. Rather, the focus lies on making the indicators easier to extract from the reduced data instead of the entire sensor data analysis. With the visualizations easier to obtain from the reduced number of features from the area and distance data, comparisons can be drawn early to detect change in signals. The interval time during this data recording was 10 minutes, so 6 values could be recorded in an hour. So, rising trends in signals can be detected on an hourly basis with 6 samples. Therefore, deviating trends in area or distance could lead to early detection of incoming defects, as is evident from the graphs in Fig. 2(b) and (c).

3.3.2. Use Case II: Defect Classification Problems

The bearing data is divided into experiments, the end results of which present defects in the bearings. The three defects along with the normal end result of one of the sensors are considered as four classes, namely, normal, inner race defect, roller element defect and outer race defect. Since the experiments have different number of samples, we consider a uniform number of samples from each class. As a result, the complete dataset consists of 2156 samples from each class and each sample has 20480 values. The total number of samples is $2156*4 = 8624$. After applying reduction, we have 2156 samples of each class with the range of values of each sample reduced to 32 from the 33 rotations. So the reduced data is of the order 2156×32. Thus, our datasets comprise of mixed sensor values ranging from the four eventual phases of the bearings.

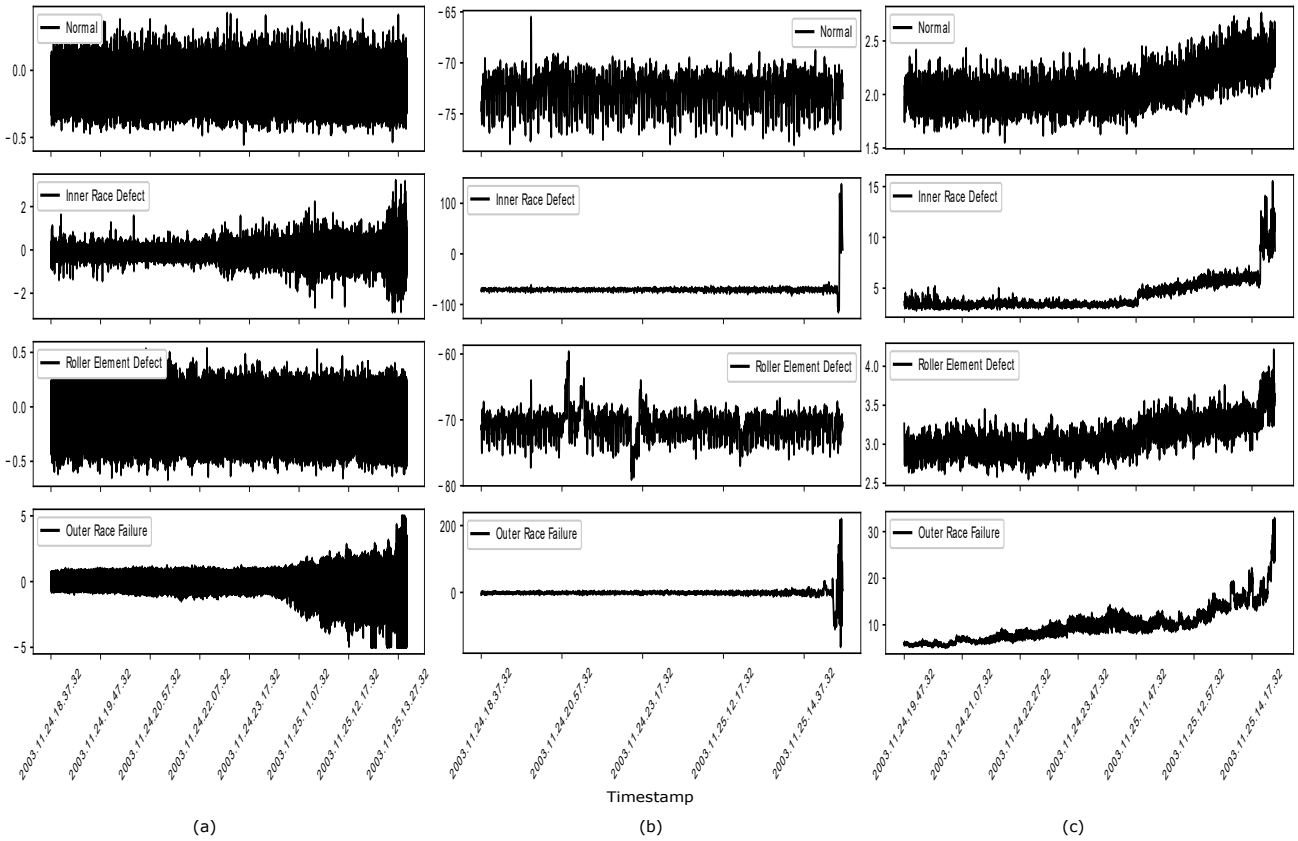

Fig. 2: (a) Actual Sensor Data in Amplitude (mV) (b) Reduced Area Data (Average area between successive packets) (c) Reduced Distance Data (Distance between successive packets)

With all the classes in one dataset, we shuffle the temporal order of the samples for the models to learn the individual sample characteristics. With the shuffled instances, if the basic constructed models are able to perform well, then many sensors exhibiting similar defects could also be included with their samples for in-depth comparisons and improvements in predictions. The objective is to identify a certain sample corresponding to a defect without depending upon its previous inputs. As such, this portrayal of time series falls into supervised classification. Such a problem was also considered by the authors in [3].

To present the applicability, only the reduced distance data is used with different neural network with varying number of layers. Each network is constructed using Keras in Python 3. The machine used is Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz. The training consists of three phases, training, validation and testing. The validation set is used to minimize the loss function, which in this case is the categorical cross-entropy. In order to change the labelled categories, we use encoders to convert to numerical classes using binary matrix encoding provided in Keras. The selection of the best model is based on the calculation of accuracy metric on the unseen test set.

The model architectures and the obtained corresponding metrics is presented in Table 1. The best performing model is with 3 layers having a test accuracy of around 95%. For industrial purposes, reduction may present an alternative approach of predicting defects with comparable accuracies as shown. However, with expanding data, the approach may need refinement of training parameters or different models.

The use cases depict the applicability of reducing time series data, which are not exhaustive. This is an extension of work from [9], so the presented examples only briefly show the possible realistic applications that can be implemented in industrial cases without excess efforts or requirements.

Table 1. Defect Classification Metrics Using Multiple Layers

Layers	Validation Accuracy	Test Accuracy
100	90.976822	90.216553
(512, 256)	95.860928	94.779581
(512, 256, 128)	95.529801	95.862335
(512, 256, 128, 64)	95.860928	94.818252
(512, 256, 128, 128, 64)	94.536424	89.365816

3.4. Blueprint Process Model Development

A process model presents a sequential outline of events and decisions during a process execution [12]. This aids in identifying process related issues to be rectified. Process models are developed either manually or using process mining techniques. Figure. 3 shows the extracted basic process model for each sample monitoring from sensors. The focus remains on presenting a basic model rather than enhancing it, since this can be done using event log generation from factory floor dynamics.

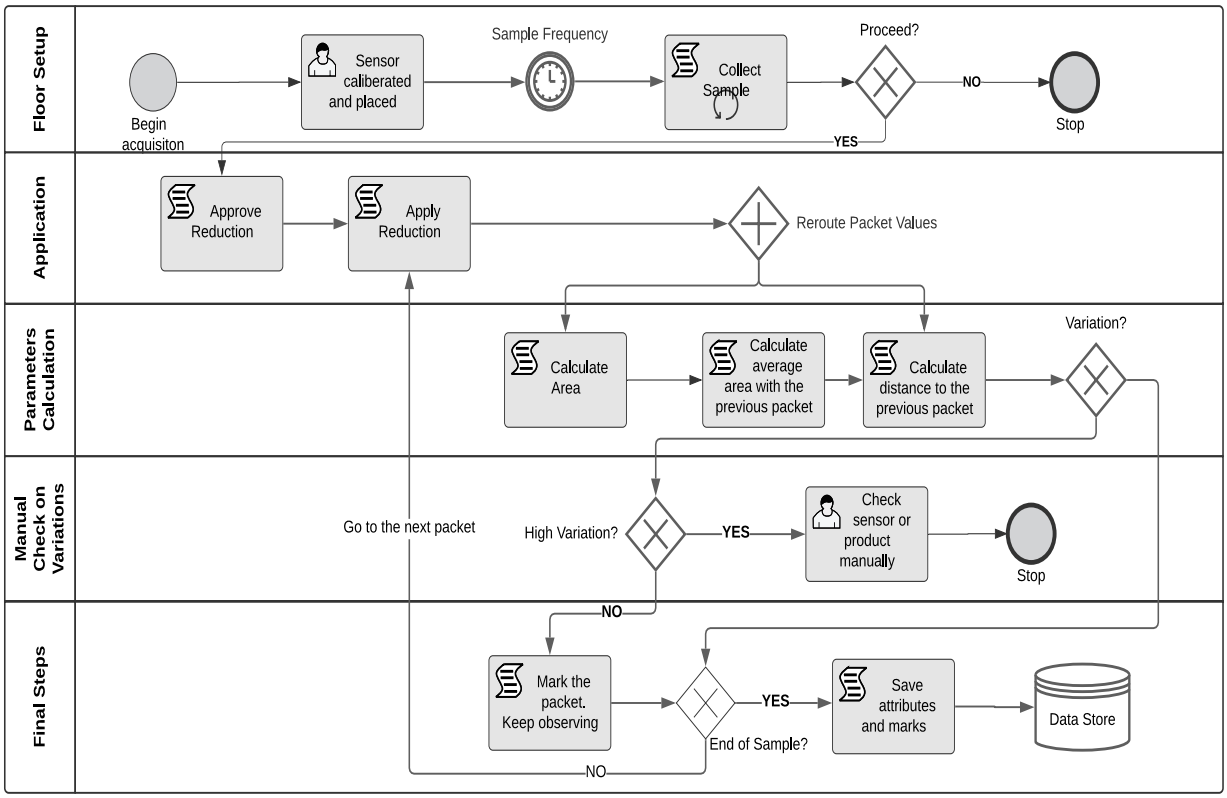

Fig. 3: Process Model for Sensor Monitoring developed using BPMN Template

Once reduction is applied, we identify events based on indicators from the reduced sensor values. Based on these, a trace of events can be extracted that can be used for marking key process instance. An example of a process instance based on a temporal sample is given below.

<(collect sample), (apply reduction), (check variations), (save sample)>

Based on such instances, the process model outlined in Fig. 3 presents a collection of all such process instances in sensor monitoring that can be sequentially executed. Some tasks are user based while others are script based showing programming agents that can execute the task without manual intervention. It depicts a methodological scenario where reduction of data can be implemented or integrated into everyday monitoring scenarios. The process shows a step-wise implementation beginning with acquisition of signals from sensors, passing through the reduction and visualization pipeline and ending with storage of marked observations or samples into designated storage. Thus, such a blueprint would help in building a flow of sequential task execution in cases of sensor monitoring. This is just an example of how to integrate reduction of data in industrial environments rather than an application of it. An in-depth process model can be developed in the context of intended use, but this is out of the scope of this paper. A common data acquisition system such as embedded PC controllers on which additional programming capabilities can be installed, should be able to incorporate the entire application field. An IoT based application with the necessary modules that can implement these cases is also presented in [9]. This process model outlines the tasks execution irrespective of systems or platforms used, thereby providing more flexibility in building applications.

4. Conclusion and Future Outlook

A scheme of reducing time series data obtained from sensors placed on bearings is presented that is applicable to any interval based time series data. Two use cases employing this reduction technique based on visual monitoring as well as in advanced analysis such as machine learning problems are also presented. Hence, the proposed technique can be handy in many industrial related problems where, if not advanced prognosis, then preliminary detection of variations in sensor behaviors can be anticipated based on the two use cases.

The presented use cases can also be combined to give a complete monitoring application. With many technical developments from advanced paradigms such as Cyber Physical Systems, Internet of Things, etc. present in current research, the process model outlined along with the use cases can be used as a basis for building such monitoring designs.

Considering the above use cases, a further approach to build reduction process that is not time-critical is in the works. With data evolving to an ever expanding volume, a methodology to carry out meaningful reduction of time series data within an acceptable time complexity is the focus of future work. It also remains to be seen whether reducing data is necessary in every case of industrial

based sensor data or not. Therefore, when to implement such a reduction is also a future direction.

References

[1] Ireneusz Czarnowski, Piotr Jędrzejowicz, and Sergio Gómez. An approach to data reduction for learning from big datasets: Integrating stacking, rotation, and agent population learning techniques. *Complexity*, 2018:7404627, 2018. ISSN 1076-2787. .

[2] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching, 2020. URL http://arxiv.org/pdf/2006.05929v2.

[3] Ran Zhang, Zhen Peng, Lifeng Wu, Beibei Yao, and Yong Guan. Fault diagnosis from raw sensor data using deep neural networks considering temporal coherence. *Sensors (Basel, Switzerland)*, 17(3), 2017. .

[4] U. Surya Kameswari and I. Ramesh Babu. Sensor data analysis and anomaly detection using predictive analytics for process industries. In *2015 IEEE Workshop on Computational Intelligence: Theories, Applications and Future Directions (WCI)*, pages 1–8. IEEE, 2015. ISBN 978-1-4673-8215-1. .

[5] Hugo Hromic, Danh Le Phuoc, Martin Serrano, Aleksandar Antonic, Ivana P. Zarko, Conor Hayes, and Stefan Decker. Real time analysis of sensor data for the internet of things by means of clustering and event processing. In *2015 IEEE International Conference on Communications (ICC)*, pages 685–691. IEEE, 2015. ISBN 978-1-4673-6432-4. .

[6] Zakarya Elaggoune, Ramdane Maamri, and Imane Boussebough. The multi-agent system solutions for big multi-sensor data management. *Journal of Ubiquitous Systems and Pervasive Networks*, 11(2):23–29, 2019. ISSN 19237332. .

[7] Muhammad Syafrudin, Ganjar Alfian, Norma Latif Fitriyani, and Jongtae Rhee. Performance analysis of iot-based sensor, big data processing, and machine learning model for real-time monitoring system in automotive manufacturing. *Sensors (Basel, Switzerland)*, 18(9), 2018. .

[8] Shang-Yi Chuang, Nilima Sahoo, Hung-Wei Lin, and Yeong-Hwa Chang. Predictive maintenance with sensor data analytics on a raspberry pi-based experimental platform. *Sensors (Basel, Switzerland)*, 19(18), 2019. .

[9] Selvine G. Mathias, Daniel Grossmann, and Tapanta Bhanja. Exploring distance based approaches for reducing sensor data in defect related prognosis. *Procedia Computer Science*, 184:614–621, 2021. ISSN 1877-0509. .

[10] Lee J, Qiu H, Yu G, Lin J, and Rexnord Technical Services. Bearing data set, 2007. URL Kaggle(https://www.kaggle.com/vinayak123tyagi/bearing-dataset).

[11] Hai Qiu, Jay Lee, Jing Lin, and Gang Yu. Wavelet filter-based weak signature detection method and its application on rolling element bearing prognostics. *Journal of Sound and Vibration*, 289(4-5):1066–1090, 2006. ISSN 0022460X. .

[12] Artem Polyvyanyy, Sergey Smirnov, and Mathias Weske. *Business Process Model Abstraction*, volume 1, pages 149–166. 12 2009. ISBN 978-3-642-00415-5. .