Predictive Maintenance Based on Machine Learning Model

Bassem Hichri, Anass Driate, Andrea Borghesi, Francesco Giovannini

Predictive Maintenance Based on Machine Learning Model*

Bassem HICHRI[1][0000−0002−0931−7341], Anass Driate[1][0000−0002−5019−6764], Andrea Borghesi[2][0000−0002−2298−2944], and Francesco Giovannini[1][0000−0002−1435−0314]

[1] GCL International, Guala Closures Group, Luxembourg
bhichri@gclinternational.com
https://www.gualaclosures.com/
[2] University of Bologna, Italy

Abstract. This paper addresses the problem of predictive maintenance in industry 4.0. Industry 4.0 revolutionized companies in the way they produce, manufacture, improve and distribute products. Industries are competing to implement and develop digital technologies driving Industry 4.0 which leads to increased automation (integration of advanced sensors, embedded software and robotics that collect and analyse data and allow for better decision making), predictive maintenance, self-optimization of process improvements and, above all, a new level of efficiencies and responsiveness to customers not previously possible. From this context the goal of the proposed work is to provide an industrial use case of machine smartifying to predict its Remaining Useful Life based on internal and external data collection and analysis using a Machine Learning algorithms. A digital Twin dashboard for real time monitoring of the machine and the result of the Machine Learning model prediction will be presented .

Keywords: Predictive Maintenance · Digital technologies · Remaining Useful Life · Machine Learning algorithms.

1 Introduction

Industrial facilities and the machinery composing them are extremely complex systems which are a fundamental parts of nowadays world. The quest for improving their management and functioning has crucial importance, with huge impacts on economy and society. Predictive maintenance is one of the key innovations enabled by the intensive exploitation of big data in Industry 4.0. Starting from real-time data harvested by embedded sensors on machineries in industrial plants, and properly gluing and fusing these data, predictive maintenance makes it possible to forecast the time to fault or the probability of fault in specific machinery components on the basis of features extracted from data. The forecasting is based on simulation and machine learning techniques that identify the most

* Supported by the EU Commission for IoTwins project number 857191.

salient features of sensor data to enable the prediction and extraction of models of machinery functioning [10].

Maintenance is a core aspect of every manufacturing process. At its most naive, it simply consists in substituting a component when it breaks (reactive maintenance), which could lead to enormous costs. A slightly better approach is to schedule maintenance regularly, according to the characteristics of the asset and related mathematical models. An increasingly popular alternative is to make use of rich historical data and Machine Learning (ML) techniques to create digital models of the industrial systems. Thanks to such models it is possible to characterize the target system's behaviour in detailed way, allowing to forecast its behaviour, to plan in advance maintenance operations and to optimize its management (i.e., with proactive workload balancing); broadly speaking, the methods combining Big Data and ML approaches fall under the umbrella of *predictive maintenance* [9].

Among the many aspects covered by predictive maintenance approaches, the Remaining Useful Life (RUL) of an asset or system is defined as the length from the current time to the end of the useful life [8]. Being able to predict this metric with sufficient reliability is among the chief research question in the domain of predictive maintenance. If the end of useful life of a component is predicted to be before the scheduled maintenance, an unexpected failure can be handled; if the scheduled maintenance is near, but the metric tells us the component is still healthy enough, more value can be extracted out of it before substitution. In recent years, Deep Learning data-driven approaches for RUL predictions have been introduced in literature and applied in practice in different contexts.

The proposed work will present online data collection from industrial plant for a selected injection moulding machine to train a Machine Learning (ML), model capable of accurately approximating the machine behavior, in particular focusing on the RUL estimation. The main contribution of this work is the creation of a *digital twin* for the industrial moulding machine using a holistic approach which integrates data from the IoT and Edge layers and uses it to train ML models for predictive maintenance, exploiting High Performance Computing resources for the computation-intensive training. The proposed approach has been implemented and deployed on a real industrial testbed.

This paper is organized as follow: Sec. provides a brief overview of related works, then Sec. 3 states the treated problem. Sec. 4 details the overall proposed architecture to estimate the RUL and the technical setup of the injection moulding machine. Sec. 5 discusses the experimental results.

2 Related Works

Industrial **fault identification** and **diagnosis** have gained a lot of interest in the last decade and with industry 4.0 revolution, many industries and research institution focused on statistical learning methods (*e.g.* probabilistic models, Bayesian networks, etc.) to develop solutions applied to diverse production areas, from electrical motors [1] to complex systems [2].

Fully supervised approaches for anomaly detection based on neural networks (NNs) have been widely discussed in recent years. Wang et al. [3] propose an approach for fault diagnosis on power systems based on sparse stacked autoencoder (SSAE) NNs. SSAE are are composed by a set of sparse autoencoders disposed in a chain-like structure, where the output of the previous autoencoder is fed as input to the following one. Siegel et al. [5] propose a supervised approach for arc-fault detection in electronic circuits for the Internet-of-Things, based on a deep NN acting as a classifier and trained on real data. In the data center context Borghesi et al. [6] proposed the usage of a semi-supervised method for anomaly detection in data centres which require only normal data and is based on an autoencoder deep NN: the NN learns the behaviour of the normal system state and is then capable to recognize it from faulty conditions due to their implicit difference (normal points are projected in different areas of the latent space w.r.t. anomalous examples). This capability of not requiring explicit labels can be a key strength in that context where labelled data are very costly to obtain and where the anomalous events are extremely rare [7].

In recent years, many Deep Learning (DL) algorithms tailored for dealing with time-series data have been applied to industrial contexts. In particular, the most widespread DL models for time-series forecasting are Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN). RNNs are endowed with a memory component which allows them to remember previous states, in contrast with standard feed-forward networks which have a combinatorial nature. In RNNs the neuron has a state which depends on the state generated in the previous time step. RNNs can be fed with sequences of variable length and can be applied in different ways, according to the relationship between the input and output RNNs have been widely used for RUL estimation in industrial settings as well [15, 13]. CNNs were originally devised for dealing with images, by working on the implicit spatial *locality* of images (pixels close to each other are likely to be correlated); 1-dimensional CNNs have been shown to effectively exploit the *temporal locality* underlying time-series data. 1D CNNs have been used in many industrial applications as well [11, 12]. More recently, a variant of CNNs called Temporal CNNs (TCNNs) [14] has emerged and used in industrial contexts, especially for dealing with multi-variate time-series data [16, 17].

3 Problem statement

The proposed use-case of Guala Closures Group (GCL) in this paper is a showcase and an application of the concept of **IoTwins** European project in the manufacturing sector. IoTwins aims at enabling SMEs in the manufacturing and facility management/service sectors to access edge enabled and cloud based big-data analysis services to create hybrid digital companions to improve their production process and optimize the management of their facilities. A production plant of GCL Group contains up to 200 machines with different purposes, brands, and type of use. With the support of the maintenance team and department managers, monitoring activities have been introduced to identify all

possible unexpected events that should occur during the production life cycle and its possible related causes. Among the various hypothetical failure events examined, it was decided to move towards a dedicated event of breakage which cyclically, but unpredictably, occurs on a defined plastic injection moulding machine. The anomaly that has been decided to prevent is an abnormal wear of the spindle bearing coating (worm screw that allows the mobile table to close by acting on the brace of the toggle). The material that comes off the bearing settles on the crests of the worm screw, damaging it.

The goal of the proposed work is to predict the occurrence of the "alarm" status during normal production runs. This will be done by developing, with the help of our project partners from University of Bologna, validating and testing a machine learning model, based on deep learning, capable of predicting the **Remaining Useful Life RUL** of the ball bearing. The model outputs the RUL in the form of a survival probability (cf. 1), with values ranging from 0 to 1, where 1 indicates that no failure will occur within now and a given time window, and 0 indicating the certainty that a failure will occur within now and a given time window.

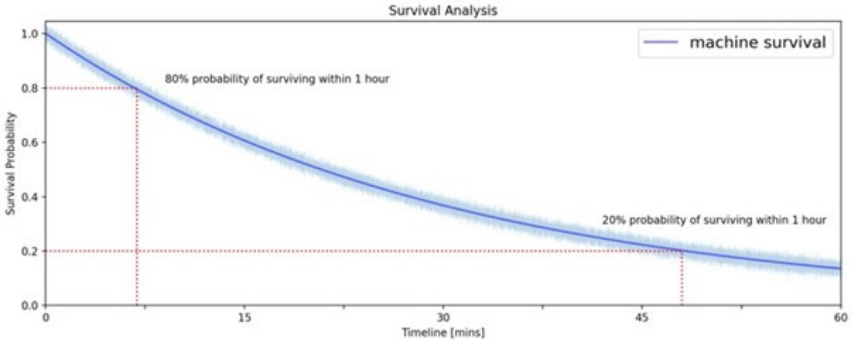


Fig. 1. Survival Probability Model Outputs

4 Overall proposed architecture

In this section we provide details on the ML model used forecast the RUL of the target industrial component to and on the overall architecture of the proposed methodology.

4.1 Machine Learning Model

The data from the sensors has the shape of a multi-variate time-series; the sampling rate is 5 minutes. We start by pre-processing it to remove clearly useless values (e.g., constant values for all the monitored period) and by normalizing it in the [0,1] range. Then we prepare the data for the actual task of RUL estimation; the key aspect is building the RUL label. We adopt the classical method of segmenting the time-series in runs-to-failure[18], i.e., periods of time where

a component starts as perfectly healthy and goes towards the end of its useful life (where RUL is 0). This can be done as we have information describing the state of the component at every time step. The next step is to create multiple sequences from the historical data (the); we chose a sequence length equal to 512 time steps (each time step is 5 minutes), corresponding to more or less 43 hours.

The RUL estimation ML model is a supervised one: it relies on the presence of a label associated with the data (time-series) collected from the target system. To effectively train the ML model the training data (the data used to teach the DL models) must contain some run-to-failure events, that is periods of time where a failure/a problem/an anomaly happened; there must be at least one run-to-failure, but a larger amount is welcomed and could greatly improve the accuracy of the service. In the absence of critical events (e.g., the target system has a lifespan of several years and no failure has yet happened), simulated or synthetic run-to-failure data could be generated. We implemented and performed experiments with several types of neural network for handling the time series and predicting the RUL (after the data have been pre-processed): RNNs, CNNs, and TCNs. After an empirical preliminary evaluation we opted for the 1D CNN as it provided the best trade-off between prediction accuracy and model inference time. In fact, it must be noted that we both want accurate measures but at the same time we prefer DL models not overly complex, as the trained DL model should continuously executed on an edge device to produce the live estimate of the RUL.

The model take as input a batches of sequences, and so 3-dimensional data (number of batches $\times$ sequence length in steps $\times$ number of features). After the input layer, there is a series of five 1D convolutional layers, each with the same composition: 128 neurons, stride and kernel size equal to 2, and batch normalization. After the convolutional layers we apply 1D max pooling, followed by a dense layer with 32 neurons. All activation functions are ReLU. The output layer has a single neuron with a linear activation function, which is common practice for regression models; the model is trained to minimize the Mean Squared Error of the predicted RUL. We used 100 training epochs and a batch size of 256; Adam optimizer has been used. The model was developed using the TensorFlow[3] Python package (v2.2).

4.2 Architecture

On-line data and off-line historical data from industrial plant and machinery are used to generate an upstream flow of data to feed the machine learning models. To realize the required upstream of data, industrial control technology and IT solutions are implemented to the manufacturing testbed.

The chosen IT architecture (cf. Fig. 2) decided to approach this project was Edge-Cloud distributed architecture, in our case and to facilitate the comprehension of this document, we divided it in 3 different levels: IoT/Premises, Edge and

[3] https://www.tensorflow.org/

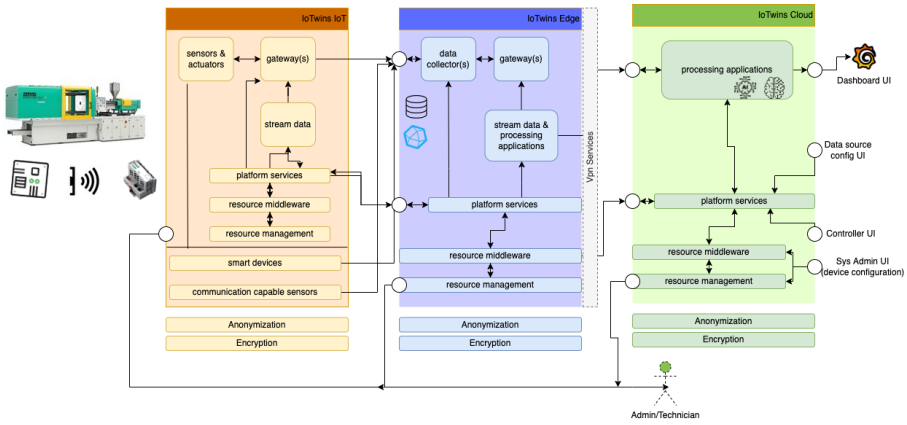


Fig. 2. Architecture Overview

Cloud. the first level consists of devices that are equipped with different kinds of sensors that capture and send information using different industrial protocols. Within the scope of this project, it has been added some critical ball bearings sensors on the injection moulding machines, measuring its vibration speed, acceleration and temperature. Between all variables, it has been identified which ones are the most responsible for indicating the overall health of the machine. Then, following the ISO 10816-3 regulation, it has been defined thresholds for these measured values, where each threshold corresponds to one of four bearing statuses – normal, pre-warning, warning, alarm. Due to the heterogeneity of the communication methodologies, emerged the necessity of a middleware communicator, that would be capable of integrating the information exposed by the machines and analysing the data. The first middleware chosen for the edge server (second level) was KepServerEX, a platform of PTC, a world leader in the world of the Internet of Things. This tool incorporates hundreds of different communication drivers and can connect directly to the machine, exposing the normalized data to the outside. We also employ an additional middleware called ThingWorx, as it is capable of processing the information received and storing it in any type of database; the database chosen for the project was InfluxDb, which is an open-source time series database (TSDB) developed by InfluxData. It is optimized for fast, high-availability storage and retrieval of time series data in fields such as operations monitoring, application metrics and Internet of Things sensor data. Fig. 3 shows the data flow of the proposed use case.

After data is processed by the edge middleware (KepServerEX, ThingWorx) and stored in InfluxDB, it is visualized in a comprehensible way using Grafana tool. Grafana is a multi-platform open source analytics and interactive visualization web application. It provides charts, graphs, and alerts for the web when connected to supported data sources. Using docker (Docker is a set of platform as a service products that use OS-level virtualization to deliver software in packages called containers. The service has both free and premium tiers. The software

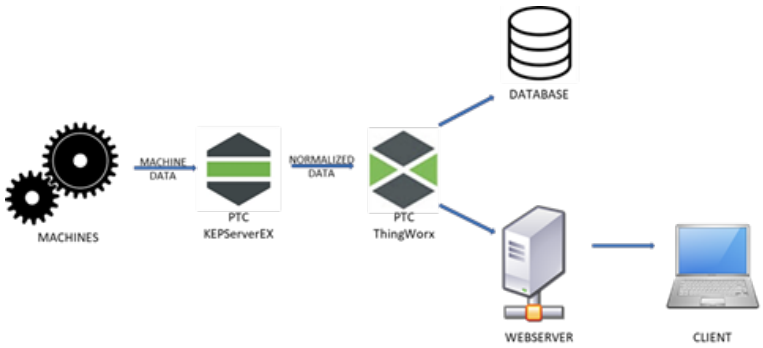


Fig. 3. Data flow of the testbed

that hosts the containers is called Docker Engine) many services are running as containers to link between the edge level and cloud level. Every 15 min data is retrieved from the InfluxDB using a specific service and reshaped to a the cloud required format using another service. Every 1 hour 4 csv files are sent to the cloud to run the ML model. For training the model we use the CINECA supercomputer. the HPC system chosen for this case, uses these information to train the model exploiting CINECA computing resources. Results of this activity are neural network models that are used to calculate RUL estimation of the injection machine component object of this use case. Every 2 months it is scheduled a re-train activity with the up-to-date information in order to provide a more efficient and reliable model. CINECA HPC also provides a specific REST API used to obtain in real time the information of the RUL expressed in hours.

4.3 Technical set up

The machine identified (cf. Fig 4) was, at its state of the art, not connectable to any platform because of the lack of technical equipment (hardware and software). With the support of the machine supplier the machine has been equipped with the interfaces to expose externally its process data and the plant was predisposed with an appropriate networking structure in order to connect our plant machines with the edge server.

Moreover, PTC KEPServerEX[4] has been installed on the edge server, this software provides a series of drivers and plugin that allow the machine to communicate with our platform. KEPServerEX is an industry connectivity platform that provides a single source of industrial automation data to all the applications. The platform design allows users to connect, manage, monitor and control different automation devices and software applications through one intuitive user interface. Connected to KEPServerEX, an instance of another software, PTC ThingWorx[5], is responsible elaborating the normalized information returned,

[4] https://www.kepware.com/en-us/products/kepserverex/
[5] https://www.ptc.com/en/products/iiot/thingworx-platform

(a) Moulding machine. (b) Integrated sensors.

Fig. 4. TestBed set up.

store and manipulate them into an historical database and send them to the webserver displaying human readable information to the end-user. The moulding machine chosen (see Figure 19) for the pilot natively provides about 2000 information tags, but just some of them are sensible information about the process. To identify the correct breakdown event, it was necessary to find out the best way to obtain this information in real time. The solution adopted was to mount on the molding machine a series of new sensors to obtain real time information. In particular:

- PLC Wago[6] this tool allows the platform to retrieve information about the electric consumption.
- Pruftechnik VIBGUARD COMPACT[7] Sensors: on Injection and Closing side. These sensors have been placed to monitor rotation, temperature, and vibration of the bearings. For the injection bearing two warning thresholds has been set to detect the breakage event to prevent. These threshold were defined based on ISO 10816-3 norm of bearing constructors.

5 Results and discussion

The validation aspect is done in one specific plant in Italy. Input data for the model is a multivariate time-series sampled every 500ms with 49 input features. Data is then grouped in batches of 15 minutes for model training. The features include:

- the electric current consumption of the machine,
- the temperature in each of the cavities of the mould,
- the velocity of the vibration of the bearing

[6] https://www.wago.com/us/discover-plcs
[7] https://www.pruftechnik.com/com/Products-and-Services/Condition-Monitoring-Systems/Online-Condition-Monitoring/Online-Condition-Monitoring-Systems/VIBGUARD-compact/

– the acceleration of the vibration of the bearing.

Figure 5 below illustrates the sample input data and an abnormal event (red line).

Fig. 5. Multivariate Features by time

Preliminary results, illustrated in Fig 6, indicate that the model predictions are in accord with the real data. Given a day in which the machine had one critical alarm at around 14:15, the model correctly begins predicting this failure at 13:30. This is illustrated in figure 34, showing a gradual decrease in model output value p from 13:30 ($p = 0.8642$) until 14:30 ($p = 0.0353$). Once the failure is resolved and the machine status goes back to Normal, the output value computed by the model increases and stabilises itself around approximately 1.

Using the machine data, the machine learning model described in section 4.1 was developed by a corresponding service in the IotWins project and sent to a demonstration/validation dashboard that has been developed in order to capture singularities that can help to improve the performance. A screenshot of this dashboard is shown in Figure 7(a).

The Dashboard was built using JAVA and JavaScript as languages and Spring and Express Js as its respective framework. There is one specific microservice in charge of handling the requests from our front-end and our edge server that will manage and redirect all of them to the respective microservice, it will also control the authorization and authentication for each request. Once the request of login is made, the application will validate that user, an access token (Json Web Token) will be created and used within every single further request.

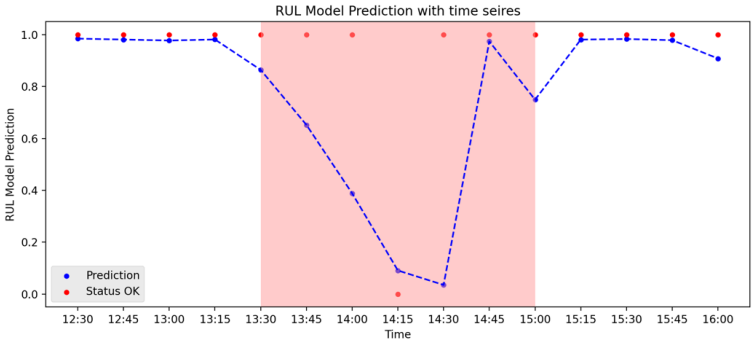
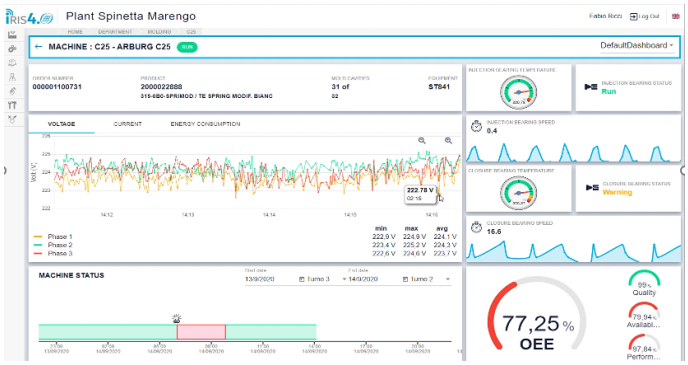


Fig. 6. Model Prediction following Bearing Status



(a) Digital Twin dashboard.

(b) RUL reporting dashboard

Fig. 7. Digital Twin and validation Dashboards

The technology used by the front-end to render all data visualisable is Angular Framework that will make requests to the server side using a representational state transfer application programming interface (REST API), it is therefore possible for the user to view information from the machines, both real time and historical data.

The platform allows users to have a corresponding digital twin of the machine with the advantage of obtaining aggregate information that can give added value to the data collected.

RUL (remaining useful life) to predict the occurrence of the "alarm" status during normal production is calculated by a corresponding service in the IotWins project and sent to the dashboard mentioned above. Using the integrated alerting features in the dashboard, thresholds can be set and defined to generate an alarm about the RUL for a specific machine. Figure 7(b).

The use of the provided cloud resources allows extending the current model by training the model on bigger datasets. This allows to improve the accuracy and enables the implementation of further use cases to predict the machine behavior. Furthermore, as with more diverse data, it is expected that also the model will become more complex, and more parameters have to be tuned. Therefore, an optimization of the ML models will be considered.

6 Conclusion

In the proposed work we connected an injection moulding machine to a custom-made data-gathering system. The collected data sources include both sensors embedded on the machine by the manufacturer, as well as other sensing equipment installed and validated by GCL. We used various communication standards – OPC/UA, CODESYS, MODBUS, OPC/DA – as well as a dedicated middleware to integrate the information exposed by the machines and analyse the data. This tool incorporates hundreds of different communication drivers and can connect directly to the machine, exposing the normalized data to the outside. In addition, we devised a SaaS architecture to handle the complete data pipeline, from acquisition to storage. Intermediate processes include validation and pretreatment. The collected data was used to train a Machine Learning model to estimate the remaining useful life (RUL) of a critical bearing in a Plastic injection moulding machine. This model will help to perform required preventing maintenance in advance to avoid any failures during production on the machine. The model runs on the Cineca HPC infrastructure. Every 15 minutes it receives fresh data from the machine and returns its prediction in the form of the Remaining Useful Life of the machine, expressed in hours.

Acknowledgements This project is funded by the EU Commission for IoTwins project number 857191.

References

1. CAI, Baoping, LIU, Yu, et XIE, Min. A dynamic-Bayesian-network-based fault diagnosis methodology considering transient and intermittent faults. IEEE Transactions on Automation Science and Engineering, 2016, vol. 14, no 1, p. 276-285.

2. CAI, Baoping, LIU, Hanlin, et XIE, Min. A real-time fault diagnosis methodology of complex systems using object-oriented Bayesian networks. Mechanical Systems and Signal Processing, 2016, vol. 80, p. 31-44.
3. WANG, Yixing, LIU, Meiqin, BAO, Zhejing, et al. Stacked sparse autoencoder with PCA and SVM for data-based line trip fault diagnosis in power systems. Neural Computing and Applications, 2019, vol. 31, no 10, p. 6719-6731.
4. SHEN, Changqing, QI, Yumei, WANG, Jun, et al. An automatic and robust features learning method for rotating machinery fault diagnosis based on contractive autoencoder. Engineering Applications of Artificial Intelligence, 2018, vol. 76, p. 170-184.
5. SIEGEL, Joshua E., PRATT, Shane, SUN, Yongbin, et al. Real-time deep neural networks for internet-enabled arc-fault detection. Engineering Applications of Artificial Intelligence, 2018, vol. 74, p. 35-42.
6. BORGHESI, Andrea, BARTOLINI, Andrea, LOMBARDI, Michele, et al. A semisupervised autoencoder-based approach for anomaly detection in high performance computing systems. Engineering Applications of Artificial Intelligence, 2019, vol. 85, p. 634-644.
7. BORGHESI, Andrea, BARTOLINI, Andrea, LOMBARDI, Michele, et al. Anomaly detection using autoencoders in high performance computing systems. In : Proceedings of the AAAI Conference on Artificial Intelligence. 2019. p. 9428-9433.
8. Si, Xiao-Sheng, et al. "Remaining useful life estimation–a review on the statistical data driven approaches." European journal of operational research 213.1 (2011): 1-14.
9. Zhang, Shen, et al. "Deep learning algorithms for bearing fault diagnostics—A comprehensive review." IEEE Access 8 (2020): 29857-29881.
10. Borghesi, Andrea, et al. "Iotwins: Design and implementation of a platform for the management of digital twins in industrial scenarios." 2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid). IEEE, 2021.
11. Ince, Turker, et al. "Real-time motor fault detection by 1-D convolutional neural networks." IEEE Transactions on Industrial Electronics 63.11 (2016): 7067-7075.
12. Zhang, Wei, et al. "A new deep learning model for fault diagnosis with good anti-noise and domain adaptation ability on raw vibration signals." Sensors 17.2 (2017): 425.
13. Bai, Shaojie, J. Zico Kolter, and Vladlen Koltun. "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling." arXiv preprint arXiv:1803.01271 (2018).
14. Oord, Aaron van den, et al. "Wavenet: A generative model for raw audio." arXiv preprint arXiv:1609.03499 (2016).
15. Guo, Liang, et al. "A recurrent neural network based health indicator for remaining useful life prediction of bearings." Neurocomputing 240 (2017): 98-109.
16. Cao, Yudong, et al. "A novel temporal convolutional network with residual self-attention mechanism for remaining useful life prediction of rolling bearings." Reliability Engineering & System Safety 215 (2021): 107813.
17. Wang, Yiwei, et al. "Temporal convolutional network with soft thresholding and attention mechanism for machinery prognostics." Journal of Manufacturing Systems 60 (2021): 512-526
18. Spiegel, Stephan, et al. "Pattern recognition and classification for multivariate time series." Proceedings of the fifth international workshop on knowledge discovery from sensor data. 2011.