Artificial Intelligence Review (2024) 57:119
https://doi.org/10.1007/s10462-024-10748-9

Artificial intelligence and edge computing for machine maintenance-review

Abubakar Bala[1] · Rahimi Zaman Jusoh A. Rashid[2] · Idris Ismail[3] · Diego Oliva[4] · Noryanti Muhammad[5] · Sadiq M. Sait[6] · Khaled A. Al-Utaibi[7] · Temitope Ibrahim Amosa[3] · Kamran Ali Memon[1]

Accepted: 6 March 2024 / Published online: 15 April 2024

Abstract
Industrial internet of things (IIoT) has ushered us into a world where most machine parts are now embedded with sensors that collect data. This huge data reservoir has enhanced data-driven diagnostics and prognoses of machine health. With technologies like cloud or centralized computing, the data could be sent to powerful remote data centers for machine health analysis using artificial intelligence (AI) tools. However, centralized computing has its own challenges, such as privacy issues, long latency, and low availability. To overcome these problems, edge computing technology was embraced. Thus, instead of moving all the data to the remote server, the data can now transition on the edge layer where certain computations are done. Thus, access to the central server is infrequent. Although placing AI on edge devices aids in fast inference, it poses new research problems, as highlighted in this paper. Moreover, the paper discusses studies that use edge computing to develop artificial intelligence-based diagnostic and prognostic techniques for industrial machines. It highlights the locations of data preprocessing, model training, and deployment. After analysis of several works, trends of the field are outlined, and finally, future research directions are elaborated

Keywords Artificial intelligence · Cloud computing · Edge computing · Fog computing · Predictive maintenance

1 Introduction

Maintenance includes an aggregate of all administrative, technical and supervision processes that ensure machines continue to run in their expected manner (Basri et al. 2017; Jianzhong et al. 2020). Maintenance generally includes methods such as inspecting, testing, servicing, overhauling, replacing, repairing, measuring, and detecting faults to avoid failures that could hinder smooth production processes (Zonta et al. 2020). Efficient maintenance is able to minimize failures and prolong useful life of machines. Historically, as shown in Fig. 1, maintenance could be categorized into corrective, preventive,

Extended author information available on the last page of the article

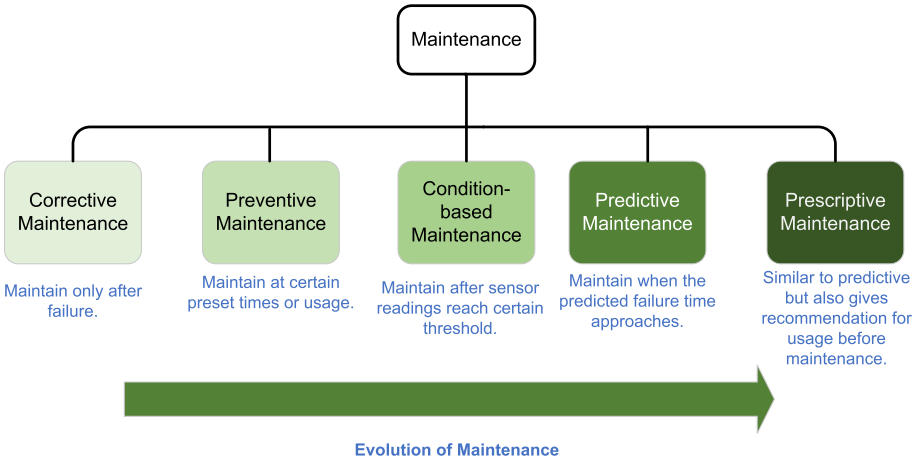


Fig. 1 The five types of maintenance approaches and their evolution

condition-based, predictive, and prescriptive, with prescriptive being the latest method (Hu et al. 2022).

In the beginning of automation, machines were simplistic. Thus, they could be operated as long as possible and maintained only after they have failed. This maintenance approach is reactive and called *corrective maintenance (CM)*. However, as machines became a little more sophisticated, corrective maintenance proved to be naive. The outcome was a considerable rise in unanticipated downtimes, resulting in severe damage to machine parts and a substantial strain on the maintenance budget.

This paved the way to what is called *preventive maintenance (PM)*, where the machine is maintained either at a regular frequency (calendar-based) or after a certain length of usage (usage-based) (Yang et al. 2019; Wang et al. 2020; Wong et al. 2022). The time or length of usage before maintenance is often the result of an educated guess. For example, it could be said that a car is due for service after it runs a certain number of kilometers. PM has the potential of reducing system downtimes, hazards due to part failures, and also maintenance cost. While this type of maintenance has its merits, it also comes with its own limitations. The machine operation hours or time set for maintenance may be inaccurate. For example, a car driven on a rough road may require early maintenance despite running a few kilometers. Thus, not maintaining the car on schedule could be dangerous. Also, the reverse may be true, when the car runs on a smooth terrain, it may not require maintenance even after reaching the preset kilometers. Hence, maintaining it before the due time amounts to resource wastage. Huang et al. (2020) have reported that in PM, majority of machines are maintained too early while they still have a large amount of their useful life available.

With advancement of electronic sensors, the condition of many parts of machines can be determined. Hence, technicians can check whether parts actually require maintenance or not. This presented another advanced proactive maintenance method called *predictive maintenance (PdM)* (Pech et al. 2021; Theissler et al. 2021; Wen et al. 2022). In PdM, data analytic tools are used to find defects and abnormalities in operations of machines so they can be maintained before they fail. PdM goes hand-in-hand with condition monitoring. In condition monitoring, sensors are deployed to

measure conditions of parts of the machines such as vibration, temperature, torque, and noise. These sensor signals are then used in PdM to determine when the system would likely fail and thus maintenance scheduled. From these definitions, it can be seen that although the PdM may require a large deployment cost because of the sensors, it offers the least maintenance cost since the usage time before maintenance can now be set more accurately with information from the sensors. In the context of our car scenario, the vehicle can be equipped with sensors that monitor certain properties of the oil, such as temperature and viscosity. By utilizing this data from the sensors, well-informed maintenance decisions can be made to precisely predict the most suitable timing for an oil change. PdM techniques have been proven to result in a staggering 900% increase in ROI, around 30% decrease in maintenance costs, 75% reduction in breakdowns, and 45% drop in downtimes (Daily and Peterson 2017; Florian et al. 2021; Zhang et al. 2019).

Some authors have another category intermediate between PM and PdM. This is called *condition-based maintenance (CBM)* (van Staden and Boute 2021; Xu et al. 2021; Rath et al. 2022). In this category, sensors are also used to monitor the conditions of the machines. However, failure predictions are not made. Maintenance alarms are raised when the sensor signals hit certain preset thresholds. The main drawback of this method is that a significant amount of degradation must have already occurred before maintenance is done.

Prescriptive maintenance (RxM) takes predictive maintenance to the next level (Gordon and Pistikopoulos (2022; Tham and Sharma 2021; Momber et al. 2022). In RxM, not only are failure events estimated, but the system is able to recommend action(s) to be taken and their corresponding consequences. For instance, when an engine is running with varying bearing temperature, PdM would be able to tell when the engine would probably fail given the temperature trend. In contrast, RxM would go further and tell us that if the engine speed is decreased to a certain level, the time before it fails may be extended. Thus, while PdM can tell the estimated usage time of machines before failure, RxM would allow us to know the effect of different operating conditions on the time to failure. The main driver of prescriptive maintenance is prescriptive analytics. This kind of analysis extends beyond predictions to exploring hypothetical events. Thus, prescriptive analytics can be regarded as a tool that uses mostly artificial intelligence techniques to provide multiple scenarios and simulations without them happening in real life (Meissner et al. 2021).

In most literature, the field of detecting, diagnosing, and predicting faults in industrial machines is called prognostics and health management (PHM) (Che et al. 2019; Fan et al. 2019; Huang et al. 2019). Table 1 compares the four maintenance schemes on some characteristics. Prescriptive maintenance is not listed since it is an extension of predictive maintenance. The table shows the strength and weakness of each method. For example, although the PdM offers more ROI, it has a high initial deployment costs. One may wonder which scheme to adopt? The answer is subjective and depends on the asset under consideration. For simplistic machines, PdM may be an overkill, while it may be the choice method for sophisticated machines like the airplane.

In data-based PHM, sensor data is collected from machines and sent to remote cloud servers for analysis. However, because of the time-critical nature of PHM, transmitting the data to the cloud leads to significant time wastage. Thus, modern methods have adopted edge computing, where the data analysis is done on devices closer to the sensors for a faster response. More details on cloud and edge computing are presented in Sect. 2.

Table 1 Different maintenance approaches and some of their properties

Features				
	Corrective	Preventive	Condition-based	Predictive
Maintenance strategy	Reactive	Proactive	Proactive	Proactive
Downtimes	✓✓✓✓	✓✓✓	✓✓	✓
Workplace hazard	✓✓✓✓	✓✓✓	✓✓	✓
Maintenance manpower	✓✓✓✓	✓✓✓	✓✓	✓
Asset lifecycle	✓	✓✓	✓✓✓	✓✓✓✓
Environment friendliness	✓	✓✓	✓✓✓	✓✓✓✓
General ROI	✓	✓✓	✓✓✓	✓✓✓✓
Initial deployment cost	✓	✓✓	✓✓✓	✓✓✓✓
Computational cost	✓	✓✓	✓✓✓	✓✓✓✓
Complexity	✓	✓✓	✓✓✓	✓✓✓✓

1.1 Related reviews

Many researchers have provided surveys on the application of edge computing technology in maintenance, for example, Hafeez et al. (2021) reviewed works that use edge computing with machine learning for predictive maintenance. However, the papers reviewed were not restricted to maintenance application. In addition, the review discussed many data preprocessing and reduction techniques that can be done on the edge. Moreover, they propose a framework where the edge could also conduct local model retraining and later merge its new model with that of the cloud. However, there was no proof of concept to see the feasibility of the architecture.

In a related review, Compare et al. (2019) presented practical challenges encountered in adopting predictive maintenance (PdM) in industry. The paper pointed out that PdM goes beyond the hardware and software used for tracking the health of machines. It involves all decisions taken from data collection to maintenance labor. They further highlighted procedures to consider before choosing the right maintenance strategy for machines. Moreover, the paper stated that it is essential that practitioners not only focus on collecting big data that fills up the memory of devices but also on fetching smart data Alsharif et al. (2020). However, the paper did not discuss maintenance in the context of edge computing.Other review works that focus on AI with edge computing for the maintenance of industrial equipment include Chatterjee and Dethlefs (2021), Lu et al. (2023), Li et al. (2022), and Ucar et al. (2024). Table 2 shows a summary of these reviews and how they compare with this paper.

To the best of our knowledge this is one of the first reviews that focuses on works that apply these three technologies, PHM, edge computing, and AI together. Moreover, the paper also details the location most researchers perform the basic tasks like data preprocessing, model training, and final model deployment. The main contributions of the paper can be summarized as follows.

- The explanation of maintenance techniques and their evolution over the years.
- The review of recent works that employ edge computing with AI for machine maintenance.

Table 2 Comparing the proposed work with existing reviews

	Focus area	Survey years
Hafeez et al. (2021)	Discussed data processing and analysis methods like sampling and data reduction at edge layer to decrease the volume of data sent to the cloud for Predictive Maintenance	2018–2020
Chatterjee and Dethlefs (2021)	Review of AI's evolution in wind energy and data-driven decision-making. Addresses current challenges including data quality, opacity of AI models, and real-time deployment issues	2012–2020
Lu et al. (2023)	Edge computing methods for signal processing-based machine fault diagnosis. Discussed lightweight algorithms and hardware platforms for IoT-based real-time processing and predictive maintenance	2018–2022
Li et al. (2022)	Presented the latest technologies in components like Edge computing, AI, and digital twins required to monitor and maintain equipment in the industrial internet	2019–2022
Ucar et al. (2024)	Reviewed the developments in AI-driven predictive maintenance. The components involved, trustworthiness, and trends	2020–2023
Our proposed work	Examines the use of edge computing for AI-driven detection, diagnostics, and prognostics in industrial machines. Discussing the location of data preprocessing, model training, deployment. Also presents the challenges of placing AI on the edge.	2017–2024

- The categorization of the works based on the location of the AI set up (model training).
- Elaboration of design issues to consider before placing AI on edge devices.
- Identification of trends and future research directions in AI-enabled edge computing for machine maintenance.

The rest of the paper is arranged as follows. Section 2 presents the edge computing paradigm. Section 3 presents a review of works that employ edge computing for machine maintenance. Section 4 of the paper discuses obstacles often encountered when placing AI on edge devices. The section also presents trends and future research directions of the field. Finally, Sect. 5 concludes the paper.

2 IIOT and edge computing

This section describes the industrial internet of things (IIoT) and edge computing technology.

2.1 Industrial internet of things

Industrial Internet of things (IIoT) originated from internet of things (IoT), which was a term first used by Kevin Ashton in 1999 while describing his research works on RFID tags (Ashton 2009; Hazra et al. 2021; Kashani et al. 2021). Accordingly, IoT is an interconnection of mobile/stationary objects, devices embedded with sensor, communication, and actuator units. These additional modules make these devices "smart," allowing them to automatically function with minimal human supervision (Sengupta et al. 2020).

Figure 2 shows a typical IoT/IIoT framework. It comprises the following four layers: perception, transport, processing, and application (Abosata et al. 2021).

The perception layer contains devices (like cameras), actuators, and sensors. Sensors collect device data and may also track environmental factors like temperature, humidity, and pressure. On the other hand, the actuators are responsible for converting digital or control signals from the higher layers into physical actions such as turning camera angles, switching devices on or off, or controlling flow valves.

The second layer is transport, also called the communication layer. It transfers data collected from the perception layer to the processing layer, often via the Internet. Typical protocols adopted are WiFi/IEEE 802.15.4, NarrowBand-IoT, Bluetooth, LoRa, and 6LoWPAN.

The third phase is the processing layer. It performs data storage, data abstraction, and data analysis. The analysis involves extracting intelligence insight from data using powerful tools like machine learning.

Finally, the application layer decodes information from the processing layer and presents it into human-readable versions such as graphs and tables for end-user consumption. It also provides record-keeping and an interface for sending signals back to devices at the perception layer.

This signal is often generated as a result of processing the data from the previous layer and is usually used to activate actuators in the perception layer. As an illustration, if the processing layer detects that a device's temperature is reaching critical levels, it can send a signal to an actuator, prompting the activation of cooling fans.

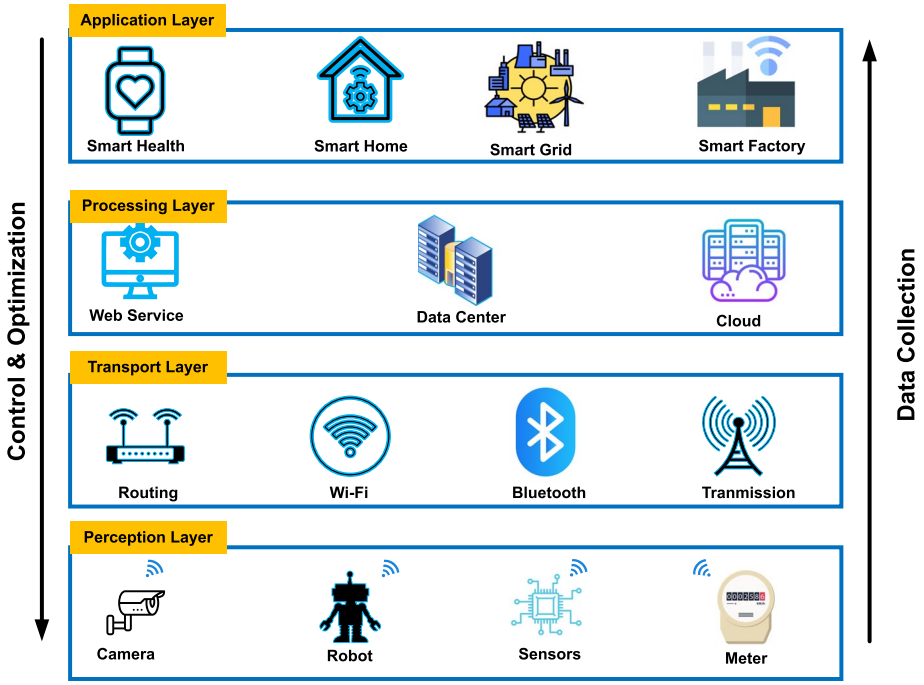


Fig. 2 An IoT/IIoT architecture (Abosata et al. 2021)

As later discussed, edge computing aims to minimize the distance traveled by data from the perception layer to the application layer and vice versa. This enables rapid decision-making, protecting human lives and avoiding multimillion-dollar losses caused by unexpected machine downtimes.

IoT has facilitated applications beyond our imaginations, today, farmers can remotely track the real-time condition of their fields from mobile devices. At their fingertips, they can monitor weather conditions, soil moisture, and even check if their farms are about to get intruded either by humans or birds (Mohamed et al. 2021; Rehman et al. 2022). Additionally, physicians can now access the health of home patients on 24/7 basis, thus reducing the number of hospital visits (Kashani et al. 2021; Bharadwaj et al. 2021).

Another area that IoT has revolutionized is home-automation (Stolojescu-Crisan et al. 2021). Here, home appliances are connected to the internet so that they can be remotely controlled and monitored. With the advent of IoT, it is now possible to effortlessly control light switches using smartphone buttons or voice commands. Additionally, smart thermostats allow us to adjust room temperatures and even provide energy usage reports. Furthermore, IoT-driven water sprinklers can efficiently water gardens at preset times, contributing to water conservation efforts (Islam et al. 2022).

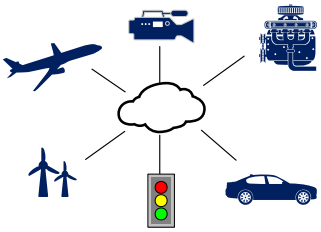
When all these technologies are moved to industrial applications, another "I" is added to IoT, making it industrial internet of things (IIoT) (Endres et al. 2019). IIoT is a connection of multiple industrial devices through a varied communication platform that enables the creation of a top-level system for collecting, monitoring, exchanging, analyzing, and providing valuable information. This system can aid industries in making

faster and smarter business decisions (Al-Turjman and Alturjman 2018). Artificial intelligence (AI) tools (Ali 2018) can use these sensor data from IIoT to help in detecting, diagnosing, and predicting faults in machines, this is the backbone of modern PdM and CBM.

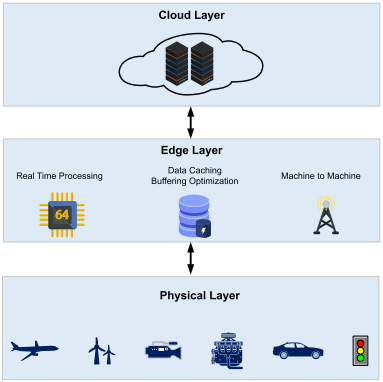
2.2 Edge computing technology

In PdM and CBM, there is often the need to analyze the sensor data collected from parts of the machine. This analysis is usually done on remote server stations. Thus, the data needs to be transmitted either via cables or the internet. These long transmission paths can lead to certain challenges such as, transmission of unwanted data, transmission delays, sending of incorrect or incomplete data due to dropped packets, and privacy concerns. Due to these and other challenges, researchers have tried to make data analysis and decisions as close as possible to the machine without the unnecessary transmission of data to remote stations. This technology is called edge computing. Edge computing is a distributed computing technology that brings applications closer to their sources of data like IoT devices and local edge servers (Cao et al. 2020; Khan et al. 2019; Satyanarayanan 2017). It earns its name "edge" from the fact that now computing resources are located at the edge of the network rather than in remote core place as in cloud computing. This offers numerous advantages such as faster insights, shorter response times, better bandwidth availability, more battery life management, and enhanced data safety and privacy (Olaniyan et al. 2018; Cao et al. 2020; Dai et al. 2019).

Figure 3 compares edge computing with centralized computing. From the figure, it can be seen that in centralized computing like cloud, all processing and analysis is done on the remote cloud, while in edge computing, the processing is moved to the proximity of data sources. The advantages of edge computing are obvious on the upstream link i.e., from IoT devices to the cloud. However, the reverse is also true. Despite the enormous computing power in cloud or datacenter, the edge can also offer certain services for downstream communication i.e., from cloud to IoT devices such as, data caching, buffering, and routing of



(a) In centralized computing like cloud computing, all processing and decisions are done in the remote cloud. The delay and vulnerability of sending data to and fro between cloud and the machines can be a major issue.



(b) In edge computing there is a platform between the cloud and the IoT devices. Critical data is now processed and analyzed closer or at the edge of the network. Only few data is sent to the cloud.

Fig. 3 Comparing edge computing architecture with typical centralized computing (Liu et al. 2019)

Table 3 Comparing centralized computing to edge computing

Features	Centralized computing	Edge computing
1. Architecture	Follows a client–server structure where computing resources and data processing are concentrated in a central location, typically in data centers or mainframes	Decentralizes computing and data processing resources by distributing them closer to the data sources and end-users
2. Latency	Since data travels back and forth between the client and the central server, latency can be higher, particularly for real-time or latency-sensitive applications	Edge computing reduces latency by processing data locally, allowing for faster response times
3. Scalability	Limited to the capacity and performance of the centralized server or data center. Adding more resources typically requires upgrading the centralized infrastructure	Enables horizontal scalability by adding more edge devices or nodes. It distributes the workload and scales the computing resources as needed without relying solely on central infrastructure
4. Security	Allows for stricter security controls since all data is stored and processed in a controlled environment. It's easier to implement centralized security measures and ensure compliance. However, an attack on the central server may kill the entire system	Poses unique security challenges due to the distributed nature of resources. Securing multiple edge devices and ensuring data integrity and confidentiality across the network becomes challenging
5. Data transfer	Large volumes of data need to be transferred between clients and the central server, which can result in higher network bandwidth requirements and potential bottlenecks	Data processing occurs closer to the source, minimizing the need for large-scale data transfers to a central server, thus, reducing network traffic and bandwidth requirements
6. Costs	Generally involves higher infrastructure and maintenance costs since it requires substantial investments in data centers and server infrastructure. However, economy of scale may also lead to reduced costs	Reduces infrastructure costs by utilizing existing edge devices and leveraging local processing capabilities. However, managing a distributed environment and deploying edge devices can introduce additional costs

information to the right machine. In addition, Table 3 compares centralized computing to edge computing based on six parameters. From the table, it can be observed that there is no clear winner between the two. For example, while edge computing offers less network latency, centralized computing may offer better security. The choice between the two will depend on the target application. Additionally, most large network systems will combine the two architectures in a hybrid form to reap the benefit of each. However, for maintenance systems, the primary optimization goals revolve around maximizing system reliability, minimizing downtime, and optimizing overall performance. These objectives are critical in ensuring the smooth operation of the system and minimizing disruptions that may lead to inefficiencies. With these objectives in mind, it is clear that PHM maintenance systems require fast response times. Regarding response times (latency), from Table 3, edge computing outperforms centralized computing, which requires data to be sent to a remote server for processing before making decisions. Moreover, centralized computing often hits a stumbling block when plants are unwilling to share their data with the remote server for proprietary or privacy reasons. Thus, to achieve fast response times and, at the same time, ensure data privacy, the architecture of choice for most systems is edge-computing, where the central server works collaboratively with edge devices to provide fast and reliable inferences (Sriram 2022).

2.3 Notes on terminologies

Here, some common terminologies used in the literature are defined.

*Fog computing:*The term fog computing was coined by Cisco Systems Inc. (Bonomi et al. 2012; Naha et al. 2018; Yousefpour et al. 2019). It refers to an architecture where there is a fog layer sandwiched between data sources and the cloud. This fog layer provides compute, networking, and storage for data. It offers the same advantages outlined in edge computing. It is hard to make a distinction between fog computing and edge computing, because they basically mean the same thing. Thus, most researchers use the terms interchangeably. This paper sticks with the name "edge computing" to mean both.

*The cloud:*In regular cloud computing, the cloud is often stationed in some remote continent and clients subscribe for services. In this paper and the context of IIoT, the term "cloud" could mean the next hop above the edge layer having more compute resources. Thus, both the edge devices and cloud devices could be in the same building or room.

3 Edge computing AI for machine maintenance

Figure 4 categorizes the edge computing-based maintenance research into three stages, which are, data preprocessing, model training and model deployment. The data preprocessing stage involves all steps executed to prepare the data for training, these include data cleaning, denoising, normalization, standardization, feature engineering, and dimensionality reduction. Most surveyed papers would carry out this operation on the edge as shown by the thick arrow to the left of the figure, while only a few do it on the cloud (Dang et al. 2021). One of the reasons for this is because these processes are not as computationally expensive as model training and thus can be performed on the edge.

In stage 2, the model is trained. The prevalent models employed in this domain can be classified into two main categories: classical and deep learning methodologies, as illustrated in Fig. 5. Classical approaches include linear regression, random forest, and

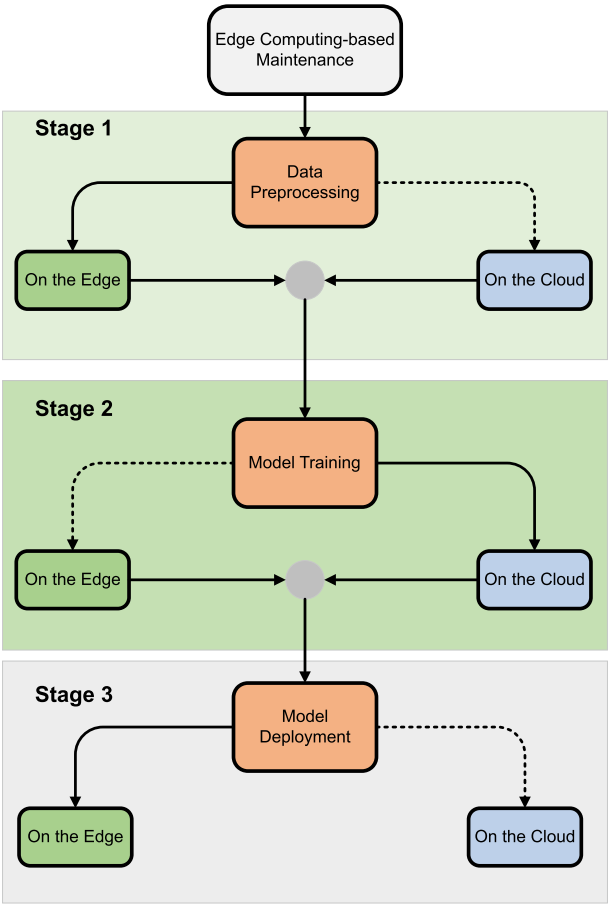


Fig. 4 Framework for categorizing edge computing-based machine maintenance

support vector machines. On the other hand, deep learning approaches consist of sophisticated models such as Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM) networks, and Gated Recurrent Units (GRU). Furthermore, some research papers propose hybrid techniques, integrating then strengths of two or more methods to achieve enhanced performance (Hsu et al. 2020).

Most researchers perform model training on the cloud because of its computational power (Wu et al. 2017). However, a few papers chose to perform model training on the edge using methods like federated learning (Imteaj et al. 2021; Yang et al. 2022).

Finally, in stage three, the model is deployed to perform the fault identification or prediction tasks. From the works surveyed, this process is mostly executed on the edge. This is partly because it is often required that the inference be as fast as possible, thus the model is moved closer to the data source. The whole process is not often static. As the machine operates, the stages are often repeated and the model is retrained to match the machine evolution or change in its environmental conditions.

The selection criteria for the papers are as follows. Scientific databases such as Google Scholar and IEE Xplore were searched with a combination of the words "edge computing,"

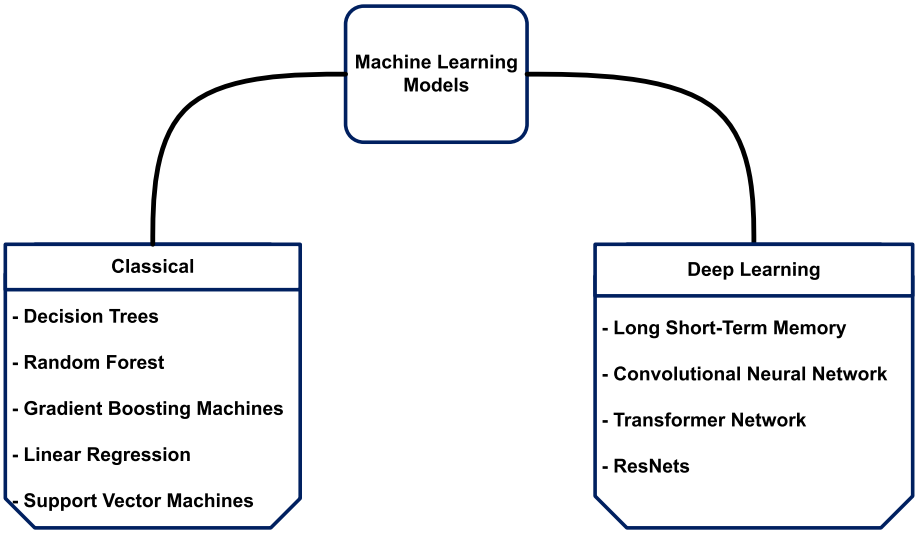


Fig. 5 Classification of common models used

"prediction," "fault detection," "fog computing," "predictive maintenance," and "condition-based maintenance." A further filtering process was performed, where the paper's abstracts were read to make sure they incorporated these three technologies in their methodologies, i.e., edge computing, PHM, and artificial intelligence. All papers that did not meet this requirement were left out.

Moreover, to review the selected research works, they are categorized into three: model training on the cloud or remote server, works that perform the training on the edge layer, and those that perform the training on both edge and cloud.

3.1 Model training on the cloud

For most works in this category, the data is collected via the edge devices and sent to the remote server for training. After the model is trained, it is deployed on the edge devices for inference.

For instance, Park et al. (2018) designed an edge computing-based maintenance technique. They termed it light-weight real-time fault detection system (LiReD), It consists of two parts, frontend and a backend. In the beginning, the frontend edge device (Raspberry Pi) takes data from machine sensors and transmits them to the backend. When enough data is sent, the backend then trains an LSTM to binarily classify faults. Subsequently, the trained model from the backend is sent to the edge device at the frontend. Thus, to classify new data from sensors, the edge device can now make classifications without sending the data to the backend. The method was tested on an industrial robot manipulator and the LSTM was compared with other baseline techniques.

In Zhang et al. (2020), the cloud trains a support vector machine (SVM) to perform condition monitoring of bearings from their vibration data and the trained model is deployed on a raspberry Pi model 4B. Similarly, Wu et al. (2017) is one of the famous works that developed an edge computing maintenance strategy. The framework emphasizes the advantages of having a hybrid of a private cloud at the edge and a public cloud in terms of data

privacy and latency reduction. The model training is conducted in the cloud and deployed on the edge BeagleBone Black device. They tested the method on health prediction of pumps in a power plant and that of a CNC cutting tool. The prediction model used is the random forest and the Predix™ model of General Electric. However, they do not provide any great detail of the random forest they used or even that of the model contained within the Predix™ software. The research also provides recommendations on the communication protocols to be used in every interface of the framework.

In a compelling research, Ren et al. (2020) put forward a cloud-edge-based RUL prediction for IIoT applications. It consists of two AI models, one at the edge and another at the cloud. Both models are an improved and lightweight versions of the temporal convolutional network (TCN) (Bai et al. 2018). The developed lightweight TCN (LTCN) had lesser parameters and training times. At the edge, the LTCN takes the preprocessed sequence data at a single sampling time and gives a rapid prediction. While in the cloud, the model takes the compressed feature extracted from the LTCN from the edge at multiple sampling times and produces a more precise and smooth prediction. After enough data (the amount is not specified in the paper), an incremental learning system situated within the cloud retrains and updates selected parameters of both LTCNs. This can be useful in avoiding data-drift (Mallick et al. 2022). They tested the method on the RUL prediction of roller bearings and compared it to LSTM (Zheng et al. 2017) and GRU (Cho et al. 2014). Results they obtained showed that their technique gave faster and more accurate results.

Additionally, Zhang and Ji (2020) developed another technique for machine maintenance. It relies on energy consumption data to detect anomalies in machine operation. Thus, at the physical layer, energy meters are installed to collect energy consumption data. This data is then preprocessed on the edge. Subsequently, it is sent to the cloud where an LSTM classifier is trained and later deployed on the edge. The method was applied for fault detection in a milling machine and the LSTM was compared with other models.

In Panicucci et al. (2020), an end-to-end edge computing-based predictive maintenance was proposed. In the work, the RUL predictor is trained in a docker container within the cloud and later deployed on the edge (Alam et al. 2018). Since the model is within a container, the RUL prediction can be carried out either on the edge or the cloud. The model options are decision trees, random forest, and gradient boosted trees. Additionally, the system has a self-assessment module to retrain the model after data drift has been detected. They tested the method for the RUL prediction of a robotic arm as a proof of concept.

In related work, Hsu et al. (2020) proposed a method that uses Raspberry Pi and NVIDIA GeForce GTX 1080 Ti GPU processor at the edge for remaining useful life (RUL) prediction of an aircraft engine. They employ machine learning methods like convolution neural network (CNN), long short-term memory (LSTM) and gated recurrent unit (GRU) for the RUL prediction. They state that the final RUL prediction results are sent to a MongoDB cloud. However, the paper falls short, it does not elaborate on how the Raspberry Pi interacts with the GPU neither does it state on which of edge device the algorithms are run.

In related work, Liang et al. (2019) also proposed a three-layered maintenance method that is powered by fog/edge computing. The terminal layer consists of the sensors and analog to digital converters (ADCs) attached to the physical machine. The collected data from the terminal layer is then sent to the fog layer. This layer does data preprocessing and fault identification based on a deployed CNN model. Finally, the cloud layer, given its high computational ability is responsible for training the CNN. Moreover, if a fault is identified on the fog layer, signals are sent to the cloud for further actions. They tested the method on a CNC machine with encouraging results.

Furthermore, Huang et al. (2021) developed an edge AI-based predictive maintenance method. It follows similar trends with other works, where the cloud does the model training and the edge carries out prepossessing and real time fault prediction based on the deployed model. They used the gradient-boosted decision tree (GBDT) model with Raspberry Pi 3B+ as the main edge device. The interesting part of the research is that as the sensor signals evolve with machine usage, the trained model in the cloud is also retrained. They applied the technique on a lithium bromide absorption chiller, which is the main unit of the central air conditioning unit used in most commercial buildings. The method was able to identify a fault one day before a human technician did.

Yu et al. (2022) proposed a three-layered framework for edge computing-based predictive maintenance. It consists of the edge layer, application, and the cloud layer. As in most other works, the edge layer uses a pre-trained model (trained in the cloud layer) for the prediction. Moreover, the edge layer also performs feature engineering and prepossessing of sensor data. Thus, only a summarized version of the data or prediction results is sent to the cloud layer. As a result, the edge layer is able to make real-time predictions without needing to send data to the cloud. On the other hand, the application layer is responsible for showing fault signatures via a dashboard. The deployed model is an auto-encoder and the technique was tested on predicting faults of a reciprocating compressor.

In Gültekin et al. (2022), another maintenance technique was developed. In the method, the model is trained offline and deployed on an NVIDIA Jetson TX2 GPU edge device. Thus, the edge device now does the preprocessing and inference in real time. The model they used is the LeNet-5 CNN with short-time fourier transform (STFT) (Wan et al. 2020). Furthermore, the approach was used for condition monitoring of an autonomous transfer vehicle. They were able to obtain a reduced bandwidth requirement of 43 folds as well as a 37 times reduction in latency. One downside of the method is that the model is not automatically updated with new data.

A similar maintenance strategy was developed by Huang et al. (2022). The method involves a cloud and edge collaboration. Like other techniques, model training is done on the cloud and then deployed at the edge. Additionally, the deployed edge model was able to identify both faults and working conditions. The authors stressed that since data is time varying, the model should not be static. Thus, two approaches of model update were outlined. The first is time-triggered, where the model is updated after a certain period has elapsed or a fixed amount of data has been collected. The second is event-triggered, where the model is updated after a specified event has occurred. Due to the difficulty in identifying the right moment of model update in the time-triggered method, they employed an event-triggered method. When the threshold is hit at the edge, a model update trigger is sent to the cloud. The model used was dictionary learning Garcia-Cardona and Wohlberg (2018) and it was tested on numerical data as well on an industrial boiling roaster and high accuracy was obtained when compared to other techniques.

On the other hand, Ren et al. (2022) proposed a cloud-edge collaborative method called label-split multiple-inputs convolutional neural network (LM-CNN). The paper highlighted that the lack and imbalance of historical data is a bottleneck to most fault detection models. Thus, the research first developed an approach that enlarges the sample space of available data and then performs data augmentation. This gave them ample data that they then used to train the ResNet-18 and ResNet-34 networks. The sample space enlargement, data augmentation and training are carried out in the cloud. In contrast, the trained model is deployed on the edge. They tested the technique for fault identification in bearings and obtained competitive results.

Similarly, Mourtzis et al. (2022) developed a framework for integrating predictive maintenance and edge computing. To leverage the advantages of the edge, Raspberry Pi nodes are installed close to the sensors. The nodes perform preprocessing of collected data and use a support vector machine (SVM) to check the existence of fault or otherwise. If there is no fault, the digital twin in the cloud remains idle. However, if a fault is detected by the edge node, the data of that node is sent to the cloud to determine the remaining useful life (RUL) of the equipment. They tested the framework on a set of refrigerators with interesting results.

3.2 Model training on the edge

As discussed in the previous section, the best place for model training or any major processing is the centralized cloud which has an enormous amount of compute resources. However, in certain situations practitioners are almost forced to train on the edge of the network. One of the reasons for this is privacy issues. Due to proprietary and privacy laws, some edge devices may not be able to share their data to the cloud for model training. To solve this issue, technologies like federated learning (FL) (Yang et al. 2019; Nguyen et al. 2021; Khan et al. 2021) are used. In FL (see Fig. 6) the model is trained on the edge devices. Rather than share data, the local models share training parameters and after training, each edge device will have the final trained model.

Federated learning and edge computing are two distinct technological concepts with different motivating factors. Edge computing was primarily designed to address latency issues by pushing data processing closer to the source, thereby reducing response times for time-sensitive applications. On the other hand, federated learning aims to tackle data privacy

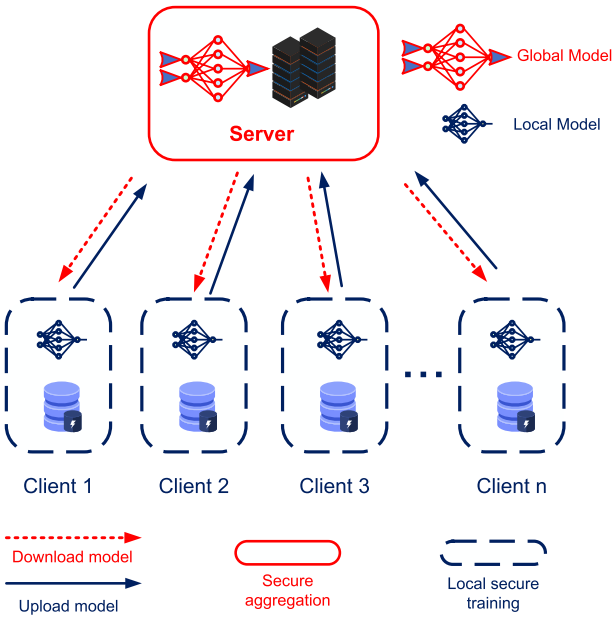


Fig. 6 A typical federated learning framework. Due to privacy, local data is not sent to the server. Local clients perform training on their data and share only model parameters via the server (Zhang and Li 2021)

concerns by enabling collaborative model training while keeping individual data decentralized and secure. Nevertheless, the two technologies can complement each other harmoniously, as federated learning can leverage the computational capabilities of edge computing to optimize its efficiency, achieving its goal of data privacy while benefiting from reduced communication overhead and low latency processing.

Some of the works that apply FL for machine maintenance include that of Qolomany et al. (2020). In this work, they developed a particle swarm optimization (PSO) (Elbes et al. 2019) technique to optimize the selection of hyperparameters of an LSTM network within the edge devices. They tested the method on the condition-based maintenance of 100 machines and compared the PSO method with other hyperparameter selection techniques. They found that their method was competitive and had less communication overhead.

In related work, Zhang et al. Zhang et al. (2020) developed a blockchain-based FL method for fault detection. The technique uses blockchain to improve the integrity of client data. Moreover, each client is rewarded with an incentive for participating in training the global model of the central server. Additionally, to mitigate the effect of data heterogeneity, they developed a new aggregating procedure based on the distance between positive and negative classes of client dataset. They tested the method on fault detection in air conditioning units and obtained interesting results.

Similarly, Zhang and Li (2021) developed an FL-based machine fault diagnostic method. Local edge devices are used to train the model. However, in each training cycle, a central server coordinates and receives local model parameters from each edge node. The server then uses a federating aggregation procedure to update the global model and send it back to the edge nodes for the next training epoch. Thus, the edge devices share model parameters but not data. At the end of training, each node would have the updated global model. The method was tested on bearing fault data using a CNN and the results proved to be competitive.

In a similar study, Zhang and Li (2022) proposed a federated learning-based industrial fault diagnostics. They state that traditional federated learning assumes that the data from different clients in the federation are independent and identically distributed (IID), which means that client data are obtained from the same machine operating in the same condition. They highlighted that this is not the case in practice. An FL model built on this assumption will not produce accurate predictions due to this domain-shift problem. Moreover, conventional transfer learning solutions to domain-shift may not work because it relies on data availability in all domains (clients), which can practically be challenging, and defeats FL's data privacy aims. Thus, they introduced a new method to solve this problem. The method uses prior distributions to help in the training. This prior distribution is simulated from client data and used in the domain adaptation rather than the raw client data. The ML model used was the basic CNN model, and they tested the method on the two machinery fault diagnostics to prove its power. The only downside of the paper is that the basic CNN was not compared with other ML models.

In Yang et al. (2021), a federated transfer learning based on averaging shared layers (FTL-ASL) was developed. The technique aimed to solve the data imbalance and the domain-shift problems. Clients in the federation have both shared and personalized parameters. Initially, a related large public data set is used to train the CNN model within the server. The pre-trained model parameters obtained are then sent to each client. Thus, client training does not start from scratch. In each federated learning iteration, the local clients update both the shared and their personal parameters using local data. Next, the clients send only the updated shared parameters back to the server for

aggregation. The process keeps repeating until the global model has converged. The technique was tested on bearing data to prove its effectiveness.

In addition, Chen et al. (2022) explained that one of the shortcomings of the federated averaging (FedAvg) algorithm is that it assumes that clients in the federation have equal contributions to the global model. However, due to domain-shift explained earlier, some clients are more important than others in most practical fault diagnostic tasks. Hence, they developed a new FL aggregation method called discrepancy-based weighted federated averaging (D-WFA). The averaging method gives larger weights to clients contributing more to the global model. The weights assigned are based on the maximum mean discrepancy (MMD) between the source and target clients. The AI model used was a feature-aligned multiscale convolutional neural network (MSCNN-FA), and it was tested on a bearing data set. The technique was found to outperform other existing aggregation methods.

Furthermore, Wang et al. (2022) developed a federated transfer learning (FTL) technique for the intelligent diagnosis of insulation defects in gas-insulated switch gears. The FTL technique involves an adversarial learning that attains domain adaptation while maintaining data privacy of clients within the federation. Additionally, they developed a novel aggregation algorithm called federated minimax (FedMM) that minimizes gradient drift due to data imbalance while improving the global model's accuracy. The model used is a four-layered second-order attention CNN. Finally, the method outperformed other FL techniques.

In related work, Li et al. (2022) developed a technique called clustering federated learning (CFL) to diagnose industrial bearings faults. The method uses a self-attention network rather than CNN. The self-attention model offers the advantage of directly extracting global and local features within the data, thus obtaining better generalization over CNN, which relies on only local features. In addition, to improve performance, K-means unsupervised learning is used to cluster client features before they are aggregated at the server. The aggregation is then carried out within each cluster. The CFL was found to be better than other aggregation methods and other models like the LSTM.

In a recent work, Li et al. (2023) proposed a new CNN-based federated transfer learning method for detecting faults in rotating machines. The approach involves two stages. In the source training stage, they generated fake data through data time-stretching. The fake data creates additional health states (labels) at the source clients to reduce decision boundaries and thus help in the subsequent domain adaptation. At the target adaptation phase, they created a prediction alignment technique that achieves knowledge transfer without requiring the source data. Moreover, they introduced an instance-level consensus scheme that introduces noise to the target data to avoid over-fitting.

In Chen et al. (2023), a new bearing RUL prediction approach based on federated learning (FL) and Taylor-expansion network pruning is proposed. The model used is the multiscale convolutional neural network with a longish full connection in the first layer (LFMCNN). The model has three units: the multiscale feature augmentation module (MFAM), the deep feature extraction module (DFEM), and the prediction module (PM). The MFAM is utilized to expand shallow features from the data stored in each client. Subsequently, the Taylor-expansion pruning criterion is used by the DFEM to delete unnecessary network nodes. In addition, each client uses its local data to reconstruct the pruned model. Next, the server uses the federated averaging (FedAvg) technique to aggregate all rebirth models into a new global model. Network pruning and rebirth occur alternatively during the model training phase to develop a compact structure. The experimental results show that the proposed strategy offers a promising solution to prognostic difficulties in data

privacy contexts. Moreover, when the model was deployed on an embedded Raspberry Pi board, it produced short inference times, proving its industrial applicability.

Additionally, Wang et al. (2023) developed another CNN-based federated transfer learning approach for fault detection in rotating machines. The method consists of three parts. First, a filter is used at the target clients to remove low-quality knowledge data from the training. Secondly, they employ batch normalized maximum mean discrepancy (BN-MMD)-based loss function during training at the target clients to reduce the domain gap between the source and target. Finally, instead of the often-used FedAvg aggregation at the central server, they used an adaptive aggregation process that assigns client weights based on their contributions. They tested the method on three benchmark datasets, and the results obtained showed its superiority.

In Du et al. (2023) proposed a transformer-based RUL predictor and tested it for the remaining useful life prediction of aircraft engines. They also developed an FL version of the technique where client data privacy is maintained. Instead of using the full structure of the transformer model, only the encoder part is used to make it light. Furthermore, they employed Bayesian optimization in selecting the hyperparameters of the encoder. The FL version was deployed on four NVIDIA Jetson Nano B01 boards. While the method outperformed many other RUL techniques, the only drawback is that, in the FL, the model starts with hyperparameters optimized with combined client data, which can lead to privacy violations.

Li and Zhao (2023) developed a zero-shot learning-based federated learning for fault diagnostics. They explain that most existing transfer learning-based FL methods assume a closed fault set, i.e., faults unavailable in one client can be found in others. This is not always true in practice, as some faults are rare, and others never exist before inference. Zero-shot learning (Feng and Zhao 2020) solves this issue by using data from seen faults and their semantic description to learn characteristics of unseen faults. The model used is the variational autoencoder (VAE). The results of testing the approach on thermal power plants proved its efficiency.

Other non-FL methods that experiment with training on the edge layer include Natesha and Guddeti (2021), which proposed an edge computing condition-based maintenance procedure. In the method, machine operation sounds are collected using audio sensors and are sent to the edge to train a machine learning binary classifier to categorize the sounds into normal or abnormal conditions. The machine learning methods used are random forest, multilayer perceptron (MLP), logistic regression, AdaBoost, and support vector machine (SVM). Both training and deployment are performed on the fog server with Intel core i5. They tested the technique on the fault identification in a pump, valve, and a fan. It was discovered that on average the AdaBoost and MLP were the best. Moreover, they compared the method with a cloud-based architecture, where the training and model deployment is done in the remote cloud. They found that their edge-based method had a way faster response time.

Similarly, Dang et al. (2021) developed a method called cloud-based digital twin for structural health monitoring (cDTSHM). To develop the digital twin of the physical structure in the cloud, they used an ensemble of a mathematical model, finite element model (FEM) and deep learning (DL). The DL method used was the ResNet-34 (Gao et al. 2021) and 1D-CNN. Both DL models rely on collected data from sensors and synthetic data generated from the FEM. Moreover, the paper highlighted that data preprocessing and training is conducted on the edge and then the model is deployed in the cloud. This contrasts with most surveyed works that do preprocessing and inference on the edge and model training in the cloud. Furthermore they incorporated a web-based application for visualization. The

method was demonstrated for monitoring the health of an experimental model of bridge and a real one with an accuracy of 92%. One downside of the research is it does not specify the type of edge device used.

Another reason to opt for training AI on the edge is when the model employed is lightweight and does not require heavy computations. For example, in Oyekanlu (2017), Onyekalu developed an edge computing-based condition monitoring system for industrial internet of thing (IIoT) applications. The system employs a lightweight SQLite database at the network's edge to collect sensor signals from machines. In the beginning, the SQLite database is populated with healthy machine data. After deployment, any incoming data to the edge is compared to the reference healthy data using magnitude and frequency analysis statistical tools. If the difference between the two signals hits certain preset thresholds, an alarm is sent to the cloud. The paper tested the method on electric motors, and a significant reduction in bandwidth usage was obtained. Similarly, in Huo et al. (2019), an edge computing-based power distribution fault detection is proposed. The method uses a wavelet transform on the edge of the network to detect faults. Additionally, Short and Twiddle (2019) proposed an edge computing-based condition monitoring and fault detection for pumps. The condition monitor and fault detection are implemented on the microcontroller edge device which hosts a digital twin of the system. Test results show that the method is robust and inexpensive.

3.3 Model training on both cloud and edge

Here, some works that do not fit into the classification above are presented. For example, in some new works, model training occurs on both the edge and cloud layers.

For instance, in Jing et al. (2022), a cloud-edge collaborative framework for remaining useful life prediction was proposed. The method consists of two distinct blueprint separable convolution neural networks (BSCNNs) (Haase and Amthor 2020), one at the edge and another at the cloud. They both cooperate with each other for enhanced prediction. The edge BSCNN is a shallow network designed for fast predictions, while the cloud BSCNN model is a deep network that provides more accurate results. To collaborate with each other, they share lower layer parameters. At first, both models are initialized, and the train data is processed (at the edge) and saved in the cloud layer. This data is used to train the cloud BSCNN model, and its trained lower layer parameters are then sent to the lower layers of the edge BSCNN model. While keeping the lower layers fixed, the edge model then also trains its deeper layers using the train data from the cloud. For inference, when raw data is obtained from the machine, it is preprocessed and denoised at the edge layer. The data is then fed to the edge BSCNN model for rapid RUL prediction. This same processed data is also sent to the cloud model for RUL prediction and both data and prediction results are stored. Since it is common for models to get stale, partly due to data-drift, the edge model needs to be updated after a while. After a certain amount of real-time data has been collected, the cloud BSCNN then shares its lower parameters, stored prediction data, and saved processed data again to the edge model for retraining (update). As with the initial stage, this update only affects the deeper layers of the edge BSCNN model. With an edge server hosting, Intel(R) Core(TM) i7-8700 CPU @3.20 GHz, 8 GB RAM, and NVIDIA Quadro P400 GPU, the framework was tested on the RUL prediction of turbofan engines (Bala et al. 2020) and compared with other methods (Li et al. 2018; Pillai and Vadakkepat 2021). The proposed technique produced faster and more accurate RUL predictions.

Similarly, the first federated learning-based RUL prediction method was developed by Guo et al. (2022). In the technique, several edge clients are used to train a global encoder and an RUL predictor without sharing data. Each edge device contains a convolutional autoencoder (CAE) Chen et al. (2017) with an encoder and a decoder. On the other hand, the cloud server contains a similar encoder and the RUL predictor. At the beginning of a training epoch, each edge CAE is trained on its local data. The trained encoders are then sent over to the cloud server for aggregation to the global encoder. The aggregation is weighted-based, with the weights depending on the performance of each encoder on validation data within the cloud. Subsequently, the global encoder is sent to each edge client to extract low-level features from their data. These features and their labels are then uploaded to the cloud and used to train the global RUL predictor. Thus, training of the RUL predictor occurs on the server. The process is repeated till the global encoder and RUL predictor are fully trained. Finally, the global encoder and RUL predictor are deployed to each client. For inference, the global encoder first extracts low-level features and sends them to the trained RUL predictor for prediction. They tested the method on the RUL prediction of a milling cutter and bearing dataset. The only drawback of the approach is that by sharing their labels with the server during training, the privacy conditions of FL may be breached.

In addition, Yu et al. (2023) developed a new federated learning-based fault diagnostics model using convolutional autoencoders CAEs. They stated that in other existing Fl methods, all the training occurs in the clients, while the server is only responsible for aggregation. This approach fails to utilize the computational capacity of the server. Hence, in their technique, a global fault diagnosis classifier (GFDC) is trained within the server. In summary, the method is as follows. First, CAEs at the client train on local data. Then, client parameters are sent to the server for aggregation using an adaptive weighting method similar to Zhang and Li (2021). The aggregated CAE is then downloaded to each client and again fed with local data for feature extraction. These features and their labels are then uploaded to the server to train the GFDC. This entire process is repeated till the GFDC produces acceptable predictions on test data. After that, the GFDC and the global CAE are deployed to the clients. Not only does their method utilize the server's computational ability, they also have limited use of bandwidth since the large GFDC does not need to be downloaded to each client at each training epoch. They tested the method on two bearing datasets to prove its effectiveness. The only loophole of the technique is that the features from clients and labels sent to the server could be used to recreate client data, defeating FL's data privacy aims. However, with encryption, this issue may be solved.

3.4 Summary

Table 4, summarizes the works surveyed in this paper. It gives the type of maintenance implemented, the edge device used, the algorithm employed, as well as the application the method was tested on. An "x" is placed in a table cell where the researchers did not specify the entry. From the table, it can be observed that most researchers perform condition-based maintenance and the most used edge device is the Raspberry Pi. Moreover, the most prevalent model used is the CNN.

Table 4 Research papers that employ edge computing for maintenance

Work	Maintenance type	Edge devices	Algorithm used	Application
Wu et al. (2017)	PdM, CBM	BeagleBone Black	Random Forest	Power plant, CNC machine
Park et al. (2018)	CBM	Raspberry Pi 3B	LSTM	Robot manipulator
Short and Twiddle (2019)	PdM, CBM	Infineon C167CS	Sliding mode observer	Water pump
Zhang et al. (2020)	CBM	Raspberry Pi 4B	SVM	Bearings
Hsu et al. (2020)	PdM	Raspberry Pi, GeForce GTX 1080 Ti GPU	CNN-GRU	Aircraft engine
Zhang and Ji (2020)	CBM	x	LSTM	Milling machine
Qolomany et al. (2020)	CBM	x	LSTM,PSO	100 machines
Ren et al. (2020)	PdM	x	LTCN	Roller bearings
Panicucci et al. (2020)	PdM	Intel i5 NUC	DT, RF, GBT	RobotBox
Liang et al. (2019)	CBM	Raspberry Pi	CNN	CNC machine
Huang et al. (2021)	PdM	Raspberry Pi 3B+	Gradient-Boosted DT (GBDT)	LiBr absorption chiller
Dang et al. (2021)	CBM	x	ResNet-34, 1D-CNN	Bridges
Ren et al. (2022)	CBM	Intel(R) i5-8500 24 GB RAM	ResNet-18,ResNet-34	Bearing Faults
Mourtzis et al. (2022)	PdM	Raspberry Pi	SVM	Refrigeration
Yu et al. (2022)	PdM	OPC servers	Auto-encoder	Reciprocating compressor
Huang et al. (2022)	CBM	PC	Dictionary Learning	Boiling roaster
Gültekin et al. (2022)	CBM	NVIDIA Jetson TX2 GPU	LeNet-5 CNN	Autonomous transfer vehicle
Huo et al. (2019)	CBM	x	Wavelet transform	Distribution power network
Zhang et al. (2020)	CBM	Raspberry Pi 3B	NN	Air-con units
Zhang and Li (2021)	CBM	x	CNN	Bearing Faults
Yang et al. (2021)	CBM	x	CNN	Bearing Faults
Zhang and Li (2022)	CBM	x	CNN	Rolling Bearing Faults
Chen et al. (2022)	CBM	x	CNN	Ball Bearing Faults
Wang et al. (2022)	CBM	x	CNN	Gas Insulated Switch Gear
Li et al. (2022)	CBM	x	Self-attention	Bearing Faults

Table 4 (continued)

Work	Maintenance type	Edge devices	Algorithm used	Application
Li et al. (2023)	CBM	x	CNN	Bearing Faults
Chen et al. (2023)	CBM	Raspberry Pi 4B	CNN	Bearing Faults
Wang et al. (2023)	CBM	x	CNN	Bearing Faults
Li et al. (2023)	CBM	x	Variational autoencoder	Thermal Plant
Du et al. (2023)	PdM	NVIDIA Jetson Nano B01 boards	Transformer	Aircraft Engines
Natesha and Guddeti (2021)	CBM	Intel Core i5 1.8 GHz	RF,LR,SVM, AdaBoost	Valve, pump, fan
Oyekanlu (2017)	PdM, CBM	x	Wavelet and frequency analytics	Electric motor

4 Trends, challenges, and future research directions

This section analyzes the patterns observed in the surveyed research papers. It also delves into the prominent challenges of deploying AI on edge devices. Furthermore, it highlights research areas that remain underexplored or have not received comprehensive investigation.

4.1 Trends

From the reviewed papers above, the following patterns can be deduced.

1. First, most papers perform data prepossessing on edge devices. This is because most tasks like data cleaning, data reductions, and feature extraction are not computationally expensive and thus can be performed on the edge.
2. Secondly, researchers perform model training on the remote cloud. The reason is that the training, especially those involving deep learning, require high computations that are better handled on the high-performance machines at the cloud datacenter.
3. The trained models are mostly deployed on the edge to make the inference as swift as possible.
4. Another trend discovered is that most of the works tested their methods on existing benchmark datasets. This can be a double-edged sword. On one hand, it shows that their algorithms are able to perform well on difficult benchmark datasets. However, it does not explain how their techniques would be applied on machinery in which there is little or no available data or how they will handle concept and data drifts.

4.2 Challenges of placing AI models on edge devices

This section highlights the challenges and adjustments needed before machine learning models are successfully deployed on edge devices.

1. *Limited computing resources:* Edge devices, like IoT devices or smartphones, typically have limited processing power, memory, and storage. Thus, most AI models need to be optimized and lightweight to run efficiently in such resource-constrained environments (Zhu et al. 2020). This becomes a huge challenge since most existing models were developed with high-performance computers, such as clouds in mind. Hence, many researchers have proposed lightweight versions of these models that could be placed on these edge devices. For instance, Nikouei et al. (2018) leverage the depthwise separable convolutional network to develop a lightweight CNN algorithm for real-time human detection in video frames. Other works that proposed lightweight models for edge devices include Almeida et al. (2022), and Huang et al. (2020).
2. *Power Conservation:* Power is another scarce resource within edge devices, especially when they are battery-powered. Thus, to conserve power, appropriate model architecture and training algorithms should be selected. Other energy-saving approaches include quantization, where the model weights and activations are represented in lower precision floating point (Coelho et al. 2021). Further methods are sparse computation, pruning of model connections (Li et al. 2020), and model distillation (Jang et al. 2020). In federated learning, methods that can preserve energy include reducing communication frequency

between clients and the central server, selective participation, offloading heavy computations to the server, and hardware optimization. In some cases, researchers may consider harvesting energy from the environment (like using solar cells) to charge the batteries of end devices (Shen et al. 2022).

3. *Data privacy and security*: Since edge devices often process sensitive data, protecting user information and ensuring data integrity is paramount. To achieve this, encryption, access control, and authentication mechanisms should be employed to protect sensitive data and ensure data integrity. In addition, secure communication protocols and data minimization techniques can further safeguard privacy. Moreover, techniques like federated learning and anonymization can also preserve user privacy during model training and data processing. Lastly, it is essential not to overlook ethical considerations and transparency in data usage to foster trust among users.
4. *Offline capability:* This refers to the ability of AI models deployed on edge devices to operate without continuous internet connectivity. Such models perform computations and decision-making processes locally on the device, avoiding reliance on cloud services. They are pre-trained and do not require constant access to new data. One way of realizing that is by using caching and local computation to enhance offline functionality while employing optimizations like model quantization and compression to reduce data and model size. The offline model can then be synchronized with cloud services during periodic internet access.
5. *Edge device heterogeneity:* When deploying machine learning models at the edge, it is crucial to consider edge device heterogeneity. This refers to the diversity of hardware, processing capabilities, memory, and connectivity options among the various edge devices in an edge computing network. Edge devices can vary significantly in processing power, memory constraints, and connectivity, which impacts the feasibility and efficiency of deploying machine learning models on them. Some edge devices might have limited processing power and memory, necessitating model optimization and quantization, while others may have custom hardware accelerators that require specialized model adaptations. One solution to this is adaptive model selection (Marco et al. 2020; Taylor et al. 2018). It involves developing multiple model variants tailored to different edge device categories based on their processing power and memory capabilities. Each edge device's resource profile is assessed during deployment, and the most appropriate model variant is selected and deployed to optimize resource efficiency, reduce latency, and enhance accuracy. This approach ensures that models match the capabilities of each device, leading to improved performance, scalability, and adaptability as new edge devices are introduced or old ones evolve.

4.3 Future research directions

The following are interesting challenges that few researchers have delved into.

1. *Data availability*: One measure issue with prognostic health management (PHM), which is a family of data-driven methods used to monitor the health of machines, is still data availability. Although there is a plethora of benchmark data that can be used to test new techniques, there is the problem of how the trained model can be used in other machinery with little or no available data. One interesting research is to see how the trained parameters on benchmark data can be fine-tuned to a new machine. This would be like what has transformed computer vision research, called transfer learning, where

the parameters trained on benchmark images are used as the initial parameters for new image sets. Although a few works have been done in this regards (Guo et al. 2018; Shao et al. 2018; Han et al. 2019; Li et al. 2019), the area still remains a gold mine for more research.

2. *Creating synthetic data*: An additional technique for solving data shortage or imbalance is the creation of artificial data. One way of doing this is from the mathematical model of the machine or after the development of its digital twin in the cloud. The generation of this digital twin for new machinery and the careful generation of balanced synthetic fault data for model training can be a fascinating research area. Moreover, recently there have been works that use generative adversarial networks (GANs) to either generate synthetic data or perform data augmentation (Gao et al. 2020; Zhang and Li 2021; Liu et al. 2022).
3. *Model update*: Another interesting research area is how an existing trained model can be retrained. This is because as machines age, the model trained for its early life may not still be applicable (concept drift). Another cause of model mismatch may be changes in the operation or even the environment in which the machine functions (data drift). Thus, where there is a need to update the model, would the model be updated based on usage (time-triggered) or would it be event-triggered? For event-triggered update methods, there is a need to develop light weight algorithms that can detect model mismatch. Also, where should the model update process take place? on the cloud or the edge? It would be interesting if edge devices can perform model updates and then later sync with the cloud.
4. *Edge communication protocols*: One significant challenge in deploying AI on edge devices is that many of these devices are powered by batteries, which are often limited in capacity and require efficient power management. To ensure optimal battery life, communication protocols between sensors and edge devices should be as lightweight as possible, minimizing energy consumption during data transmission. Numerous wireless and wired communication protocols exist, each with its strengths and weaknesses. However, with the rise of edge computing and the need for energy-efficient communication, there is a growing interest in research and development of new, robust protocols tailored specifically for edge devices. These protocols should prioritize energy efficiency and responsiveness, considering the constraints imposed by limited battery power. Another interesting area of research lies in the fusion of existing communication protocols into stronger hybrids. With this, researchers can combine the best features of multiple protocols to create new solutions that offer a balance between performance, reliability, and power efficiency. Hybrid protocols can leverage the strengths of different communication methods, such as Wi-Fi, Bluetooth, Zigbee, LoRa, or cellular networks, to achieve optimal performance in diverse edge computing environments.
5. *Light-weight algorithms*: It is true that certain problems require complex models for them to be solved. However, it is human nature to be amazed by the complex, a fallacy known as complexity bias. This makes humans think subconsciously that the complex is always better. This has flooded AI research, and journal reviewers are always looking to accept the next complex "novel" method. This has almost forced authors to write complicated methods or even cocktails known as "ensembles" and "hybrids," involving intricate math and jargon that only the first author of the paper can fully understand. Most researchers fail to try out simpler models, which often perform better, but instead jump to the complex. Authors of this paper have a different opinion, complex is not always better, and "simplicity is the ultimate sophistication." If you go down history, it is the simple machines, such as the AK-47 rifle and the internal combustion engine

that have endured the test of time. Complexity should be added only when necessary, as it seems that the overly complex only remains in theory. Thus, to fully democratize AI and move them beyond the research papers they are written on, researchers should place emphasis on simple and light methods that work, and hopefully, reviewers will understand. Thus, the creation of these simple models would enhance the deployment of edge AI in PHM. Due to their sizes and location, edge devices are known to be resource constraint in terms of memory, computation, energy, and bandwidth. Thus, as AI keeps moving closer to the edge, researchers need to develop light-weight versions of algorithms that can be hosted on these constraint edge devices and still provide real-time results. This has been a less explored area. One application is light-weight algorithms for preprocessing and summarizing big data (Rani et al. 2018; Xu et al. 2019; Azar et al. 2019). Another area of future research is light weight models, such as just another network (JANET) Van Der Westhuizen and Lasenby (2018) which is a light version of LSTM, lightweight recurrent neural network (LLRNN) Liu et al. (2019) and other tiny machine learning tinyML methods (Dutta and Bharali 2021; Signoretti et al. 2021; Asutkar et al. 2022). An additional less ventured field is model initialization and hyperparameter optimization using light weight algorithms such as light and faster versions of heuristics and metaheuristics (Palani et al. 2019).

4.4 Limitations of AI-based maintenance systems

Despite the enormous potential of AI in the maintenance of industrial equipment mentioned in the pages above, it has limitations. Although some of these have been stated as research gaps in Sect. 4.3, they can be the hurdles that may make AI-based maintenance fail. These issues include data quality and availability, making it hard for the models to learn accurately. Additionally, understanding how AI arrives at decisions can be difficult, as it often operates like a black box. This can be a significant issue since most stakeholders want to know how AI arrives at its decisions (explainable AI). Moreover, scaling across different systems and integrating with existing legacy ones can be challenging. There are also concerns about biases creeping into the algorithms, leading to unfair or discriminatory responses. In addition, ensuring the security and privacy of sensitive data can be daunting. Furthermore, getting humans and AI to work collaboratively can pose a significant challenge. To overcome these obstacles and make AI-based maintenance systems hassle-free, a collaborative effort is required that involves combining expertise in AI, data management, ethics, and human–machine interaction.

5 Conclusions

This paper reviewed studies that employ artificial intelligence (AI) with edge computing for fault detection, diagnosis, and prognosis of industrial machines. The works were classified into three based on where they perform model training. The first is the traditional approach. Most of the reviewed works fall into this class. Here, the model training occurs exclusively in the cloud, and subsequently, the trained model is deployed on the edge. This happens to be the most logical way to go about it since the cloud has huge computing resources suited for training. In the second class, the model is trained within the edge layer. This is done partly due to data privacy, and local data is not sent to the cloud. The cloud

only serves as a coordinator for the training process. In the final category, both the cloud and edge participate in training the model.

In addition, the paper also presented the most common challenges faced when placing AI models on edge devices. Finally, the paper presented future research directions. These areas include the generation of synthetic data, application of transfer learning, development of lightweight algorithms for training and hyperparameter optimization, and proposing new and secure communication protocols for edge devices.

Acknowledgements This work was supported by Universiti Teknologi PETRONAS (UTP), Malaysia, YUTP Grant 015LC0-375.

Author contributions AB, TIA, II and RZJAR wrote the paper draft, with AB also drawing all figures. NM, SMS, DO, and KALU helped to find more references and edited and polished the draft. KAM was also engaged in the process. He found more references, edited the paper, and helped to draft the response to reviewers' comments.

Declarations

Competing interests The authors declare no competing interests.

References

Abosata N, Al-Rubaye S, Inalhan G, Emmanouilidis C (2021) Internet of things for system integrity: a comprehensive survey on security, attacks and countermeasures for industrial applications. Sensors 21(11):3654

Al-Turjman F, Alturjman S (2018) Context-sensitive access in industrial internet of things (IIoT) healthcare applications. IEEE Trans Ind Inf 14(6):2736–2744

Alam M, Rufino J, Ferreira J, Ahmed SH, Shah N, Chen Y (2018) Orchestration of microservices for IoT using docker and edge computing. IEEE Commun Mag 56(9):118–123

Ali YH (2018) Artificial intelligence application in machine condition monitoring and fault diagnosis. Artif Intell. https://doi.org/10.5772/intechopen.74932

Almeida JS, Huang C, Nogueira FG, Bhatia S, de Albuquerque VHC (2022) Edgefiresmoke: a novel lightweight CNN model for real-time video fire-smoke detection. IEEE Trans Ind Inf 18(11):7889–7898

Alsharif MH, Kelechi AH, Yahya K, Chaudhry SA (2020) Machine learning algorithms for smart data analysis in internet of things environment: taxonomies and research trends. Symmetry 12(1):88

Ashton K et al (2009) That 'internet of things' thing. RFID J 22(7):97–114

Asutkar S, Chalke C, Shivgan K, Tallur S (2022) TinyML-enabled edge implementation of transfer learning framework for domain generalization in machine fault diagnosis. Expert Syst Appl 213:119016

Azar J, Makhoul A, Barhamgi M, Couturier R (2019) An energy efficient IoT data compression approach for edge machine learning. Future Gen Comput Syst 96:168–175

Bai S, Kolter JZ, Koltun V (2018) An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv preprint arXiv:1803.01271

Bala A, Ismail I, Ibrahim R, Sait SM, Oliva D (2020) An improved grasshopper optimization algorithm based echo state network for predicting faults in airplane engines. IEEE Access 8:159773–159789

Basri EI, Razak IHA, Ab-Samat H, Kamaruddin S (2017) Preventive maintenance (PM) planning: a review. J Qual Mainten Eng. https://doi.org/10.1108/JQME-04-2016-0014

Bharadwaj HK, Agarwal A, Chamola V, Lakkaniga NR, Hassija V, Guizani M, Sikdar B (2021) A review on the role of machine learning in enabling IoT based healthcare applications. IEEE Access 9:38859–38890
Bonomi F, Milito R, Zhu J, Addepalli S (2012) Fog computing and its role in the internet of things. In: Proceedings of the first edition of the MCC workshop on mobile cloud computing, pp. 13–16
Cao K, Liu Y, Meng G, Sun Q (2020) An overview on edge computing research. IEEE Access 8:85714–85728
Chatterjee J, Dethlefs N (2021) Scientometric review of artificial intelligence for operations & maintenance of wind turbines: the past, present and future. Renew Sustain Energy Rev 144:111051
Che C, Wang H, Fu Q, Ni X (2019) Combining multiple deep learning algorithms for prognostic and health management of aircraft. Aerosp Sci Technol 94:105423
Chen M, Shi X, Zhang Y, Wu D, Guizani M (2017) Deep feature learning for medical image analysis with convolutional autoencoder neural network. IEEE Trans Big Data 7(4):750–758
Chen J, Li J, Huang R, Yue K, Chen Z, Li W (2022) Federated transfer learning for bearing fault diagnosis with discrepancy-based weighted federated averaging. IEEE Trans Instrum Meas 71:1–11
Chen X, Wang H, Lu S, Yan R (2023) Bearing remaining useful life prediction using federated learning with Taylor-expansion network pruning. IEEE Trans Instrum Meas 72:1–10
Cho K, Van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y (2014) Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078
Coelho CN Jr, Kuusela A, Li S, Zhuang H, Ngadiuba J, Aarrestad TK, Loncar V, Pierini M, Pol AA, Summers S (2021) Automatic heterogeneous quantization of deep neural networks for low-latency inference on the edge for particle detectors. Nat Mach Intell 3(8):675–686
Compare M, Baraldi P, Zio E (2019) Challenges to IoT-enabled predictive maintenance for industry 4.0. IEEE Internet Things J 7(5):4585–4597
Dai W, Nishi H, Vyatkin V, Huang V, Shi Y, Guan X (2019) Industrial edge computing: enabling embedded intelligence. IEEE Ind Electron Mag 13(4):48–56
Daily J, Peterson J (2017). In: Richter K, Walther J (eds) Predictive maintenance: how big data analysis can improve maintenance. Springer, Cham, pp 267–278
Dang HV, Tatipamula M, Nguyen HX (2021) Cloud-based digital twinning for structural health monitoring using deep learning. IEEE Trans Ind Inf 18(6):3820–3830
Du NH, Long NH, Ha KN, Hoang NV, Huong TT, Tran KP (2023) Trans-lighter: a light-weight federated learning-based architecture for remaining useful lifetime prediction. Comput Ind 148:103888
Dutta L, Bharali S (2021) TinyML meets IoT: a comprehensive survey. Internet Things 16:100461
Elbes M, Alzubi S, Kanan T, Al-Fuqaha A, Hawashin B (2019) A survey on particle swarm optimization with emphasis on engineering and network applications. Evol Intel 12(2):113–129
Endres H, Indulska M, Ghosh A, Baiyere A, Broser S (2019) Industrial internet of things (IIoT) business model classification. In: 40th International Conference on Information Systems, ICIS 2019, p. 2988. Association for Information Systems. AIS Electronic Library (AISeL)
Fan J, Fan J, Liu F, Qu J, Li R (2019) A novel machine learning method based approach for Li-ion battery prognostic and health management. IEEE Access 7:160043–160061
Feng L, Zhao C (2020) Fault description based attribute transfer for zero-sample industrial fault diagnosis. IEEE Trans Ind Inf 17(3):1852–1862
Florian E, Sgarbossa F, Zennaro I (2021) Machine learning-based predictive maintenance: acost-oriented model for implementation. Int J Prod Econ 236:108114
Gao X, Deng F, Yue X (2020) Data augmentation in fault diagnosis based on the Wasserstein generative adversarial network with gradient penalty. Neurocomputing 396:487–494
Gao M, Qi D, Mu H, Chen J (2021) A transfer residual neural network based on ResNet-34 for detection of wood knot defects. Forests 12(2):212
Garcia-Cardona C, Wohlberg B (2018) Convolutional dictionary learning: a comparative review and new algorithms. IEEE Trans Comput Imaging 4(3):366–381
Gordon CA, Pistikopoulos EN (2022) Data-driven prescriptive maintenance toward fault-tolerant multiparametric control. AIChE J 68(6):17489
Guo L, Lei Y, Xing S, Yan T, Li N (2018) Deep convolutional transfer learning network: a new method for intelligent fault diagnosis of machines with unlabeled data. IEEE Trans Ind Electron 66(9):7316–7325
Guo L, Yu Y, Qian M, Zhang R, Gao H, Cheng Z (2022) FedRUL: a new federated learning method for edge-cloud collaboration based remaining useful life prediction of machines. IEEE/ASME Trans Mechatron 28:350
Gültekin Ö, Cinar E, Özkan K et al (2022) Real-time fault detection and condition monitoring for industrial autonomous transfer vehicles utilizing edge artificial intelligence. Sensors 22(9):3208

Haase D, Amthor M (2020) Rethinking depthwise separable convolutions: how intra-kernel correlations lead to improved mobilenets. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 14600–14609
Hafeez T, Xu L, Mcardle G (2021) Edge intelligence for data handling and predictive maintenance in IIoT. IEEE Access 9:49355–49371
Han T, Liu C, Yang W, Jiang D (2019) Learning transferable features in deep convolutional neural networks for diagnosing unseen machine conditions. ISA Trans 93:341–353
Hazra A, Adhikari M, Amgoth T, Srirama SN (2021) A comprehensive survey on interoperability for IIoT: taxonomy, standards, and future directions. ACM Comput Surv 55(1):1–35
Hsu H-Y, Srivastava G, Wu H-T, Chen M-Y (2020) Remaining useful life prediction based on state assessment using edge computing on deep learning. Comput Commun 160:91–100
Hu Y, Miao X, Si Y, Pan E, Zio E (2022) Prognostics and health management: a review from the perspectives of design, development and decision. Reliab Eng Syst Saf 217:108063
Huang Y, Tang Y, Van Zwieten J, Liu J, Xiao X (2019) An adversarial learning approach for machine prognostic health management. In: 2019 international conference on High Performance Big Data and Intelligent Systems (HPBD &IS), pp. 163–168. IEEE
Huang J, Chang Q, Arinez J (2020) Deep reinforcement learning based preventive maintenance policy for serial production lines. Expert Syst Appl 160:113701
Huang Y, Qiao X, Ren P, Liu L, Pu C, Dustdar S, Chen J (2020) A lightweight collaborative deep neural network for the mobile web in edge cloud. IEEE Trans Mob Comput 21(7):2289–2305
Huang H, Yang L, Wang Y, Xu X, Lu Y (2021) Digital twin-driven online anomaly detection for an automation system based on edge intelligence. J Manuf Syst 59:138–150
Huang K, Tao Z, Wang C, Guo T, Yang C, Gui W (2022) Cloud-edge collaborative method for industrial process monitoring based on error-triggered dictionary learning. IEEE Trans Ind Inf 18:1
Huo W, Liu F, Wang L, Jin Y, Wang L (2019) Research on distributed power distribution fault detection based on edge computing. IEEE Access 8:24643–24652
Imteaj A, Thakker U, Wang S, Li J, Amini MH (2021) A survey on federated learning for resource-constrained IoT devices. IEEE Internet Things J 9(1):1–24
Islam R, Rahman MW, Rubaiat R, Hasan MM, Reza MM, Rahman MM (2022) Lora and server-based home automation using the internet of things (IoT). J King Saud Univ 34(6):3703–3712
Jang I, Kim H, Lee D, Son Y-S, Kim S (2020) Knowledge transfer for on-device deep reinforcement learning in resource constrained edge computing systems. IEEE Access 8:146588–146597
Jianzhong S, Fangyuan W, Shungang N (2020) Aircraft air conditioning system health state estimation and prediction for predictive maintenance. Chin J Aeronaut 33(3):947–955
Jing T, Tian X, Hu H, Ma L (2022) Deep learning-based cloud-edge collaboration framework for remaining useful life prediction of machinery. IEEE Trans Ind Inf 18(10):7208–7218
Kashani MH, Madanipour M, Nikravan M, Asghari P, Mahdipour E (2021) A systematic review of IoT in healthcare: applications, techniques, and trends. J Netw Comput Appl 192:103164
Khan WZ, Ahmed E, Hakak S, Yaqoob I, Ahmed A (2019) Edge computing: a survey. Future Gen Comput Syst 97:219–235
Khan LU, Saad W, Han Z, Hossain E, Hong CS (2021) Federated learning for internet of things: recent advances, taxonomy, and open challenges. IEEE Commun Surv Tutor 23:1759
Li B, Zhao C (2023) Federated zero-shot industrial fault diagnosis with cloud-shared semantic knowledge base. IEEE Internet Things J 10:11619
Li X, Ding Q, Sun J-Q (2018) Remaining useful life estimation in prognostics using deep convolution neural networks. Reliab Eng Syst Saf 172:1–11
Li G, Ma X, Wang X, Liu L, Xue J, Feng X (2020) Fusion-catalyzed pruning for optimizing deep learning on intelligent edge devices. IEEE Trans Comput Aided Des Integr Circuits Syst 39(11):3614–3626
Li Q, Yang Y, Jiang P (2022) Remote monitoring and maintenance for equipment and production lines on industrial internet: a literature review. Machines 11(1):12
Li W, Yang W, Jin G, Chen J, Li J, Huang R, Chen Z (2022) Clustering federated learning for bearing fault diagnosis in aerospace applications with a self-attention mechanism. Aerospace 9(9):516
Li X, Zhang W, Ding Q, Li X (2019) Diagnosing rotating machines with weakly supervised data using deep transfer learning. IEEE Trans Ind Inf 16(3):1688–1697
Li X, Zhang C, Li X, Zhang W (2023) Federated transfer learning in fault diagnosis under data privacy with target self-adaptation. J Manuf Syst 68:523–535
Liang YC, Li WD, Lu X, Wang S (2019). In: Li W, Liang Y, Wang S (eds) Fog computing and convolutional neural network enabled prognosis for machining process optimization. Springer, Cham, pp 13–35

Liu W, Guo P, Ye L (2019) A low-delay lightweight recurrent neural network (LLRNN) for rotating machinery fault diagnosis. Sensors 19(14):3109
Liu F, Tang G, Li Y, Cai Z, Zhang X, Zhou T (2019) A survey on edge computing systems and tools. Proc IEEE 107(8):1537–1562
Liu S, Jiang H, Wu Z, Li X (2022) Data synthesis using deep feature enhanced generative adversarial networks for rolling bearing imbalanced fault diagnosis. Mech Syst Signal Process 163:108139
Lu S, Lu J, An K, Wang X, He Q (2023) Edge computing on iot for machine signal processing and fault diagnosis: a review. IEEE Internet Things J 99:1
Mallick A, Hsieh K, Arzani B, Joshi G (2022) Matchmaker: data drift mitigation in machine learning for large-scale systems. Proc Mach Learn Syst 4:77–94
Marco VS, Taylor B, Wang Z, Elkhatib Y (2020) Optimizing deep learning inference on embedded systems through adaptive model selection. ACM Trans Embed Comput Syst 19(1):1–28
Meissner R, Rahn A, Wicke K (2021) Developing prescriptive maintenance strategies in the aviation industry based on a discrete-event simulation framework for post-prognostics decision making. Reliab Eng Syst Saf 214:107812
Mohamed ES, Belal A, Abd-Elmabod SK, El-Shirbeny MA, Gad A, Zahran MB (2021) Smart farming for improving agricultural management. Egypt J Remote Sens Space Sci 24:971
Momber AW, Möller T, Langenkämper D, Nattkemper TW, Brün D (2022) A digital twin concept for the prescriptive maintenance of protective coating systems on wind turbine structures. Wind Eng 46(3):949–971
Mourtzis D, Angelopoulos J, Panopoulos N (2022) Design and development of an edge-computing platform towards 5G technology adoption for improving equipment predictive maintenance. Proc Comput Sci 200:611–619
Naha RK, Garg S, Georgakopoulos D, Jayaraman PP, Gao L, Xiang Y, Ranjan R (2018) Fog computing: survey of trends, architectures, requirements, and research directions. IEEE Access 6:47980–48009
Natesha B, Guddeti RMR (2021) Fog-based intelligent machine malfunction monitoring system for industry 4.0. IEEE Trans Ind Inf 17(12):7923–7932
Nguyen DC, Ding M, Pathirana PN, Seneviratne A, Li J, Poor HV (2021) Federated learning for internet of things: a comprehensive survey. IEEE Commun Surv Tutor 23(3):1622–1658
Nikouei SY, Chen Y, Song S, Xu R, Choi B-Y, Faughnan TR (2018) Real-time human detection as an edge service enabled by a lightweight CNN. In: 2018 IEEE international conference on Edge Computing (EDGE), pp. 125–129. IEEE
Olaniyan R, Fadahunsi O, Maheswaran M, Zhani MF (2018) Opportunistic edge computing: concepts, opportunities and research challenges. Future Gen Comput Syst 89:633–645
Oyekanlu E (2017) Predictive edge computing for time series of industrial IoT and large scale critical infrastructure based on open-source software analytic of big data. In: 2017 IEEE international conference on Big Data (Big Data), pp. 1663–1669. IEEE
Palani U, Suresh K, Nachiappan A (2019) Mobility prediction in mobile ad hoc networks using eye of coverage approach. Clust Comput 22(6):14991–14998
Panicucci S, Nikolakis N, Cerquitelli T, Ventura F, Proto S, Macii E, Makris S, Bowden D, Becker P, O'Mahony N et al (2020) A cloud-to-edge approach to support predictive analytics in robotics industry. Electronics 9(3):492
Park D, Kim S, An Y, Jung J-Y (2018) LiReD: a light-weight real-time fault detection system for edge computing using LSTM recurrent neural networks. Sensors 18(7):2110
Pech M, Vrchota J, Bednář J (2021) Predictive maintenance and intelligent sensors in smart factory. Sensors 21(4):1470
Pillai S, Vadakkepat P (2021) Two stage deep learning for prognostics using multi-loss encoder and convolutional composite features. Expert Syst Appl 171:114569
Qolomany B, Ahmad K, Al-Fuqaha A, Qadir J (2020) Particle swarm optimized federated learning for industrial IoT and smart city services. In: GLOBECOM 2020-2020 IEEE Global communications conference, pp. 1–6. IEEE
Rani M, Dhok SB, Deshmukh RB (2018) A systematic review of compressive sensing: concepts, implementations and applications. IEEE Access 6:4875–4894
Rath N, Mishra R, Kushari A (2022) Aero engine health monitoring, diagnostics and prognostics for condition-based maintenance: an overview. Int J Turbo Jet-Engines
Rehman A, Saba T, Kashif M, Fati SM, Bahaj SA, Chaudhry H (2022) A revisit of internet of things technologies for monitoring and control strategies in smart agriculture. Agronomy 12(1):127
Ren L, Jia Z, Wang T, Ma Y, Wang L (2022) LM-CNN: a cloud-edge collaborative method for adaptive fault diagnosis with label sampling space enlarging. IEEE Trans Ind Inf 18:9057

Ren L, Liu Y, Wang X, Lü J, Deen MJ (2020) Cloud-edge-based lightweight temporal convolutional networks for remaining useful life prediction in IIoT. IEEE Internet Things J 8(16):12578–12587
Satyanarayanan M (2017) The emergence of edge computing. Computer 50(1):30–39
Sengupta J, Ruj S, Bit SD (2020) A comprehensive survey on attacks, security issues and blockchain solutions for IoT and IIoT. J Netw Comput Appl 149:102481
Shao S, McAleer S, Yan R, Baldi P (2018) Highly accurate machine fault diagnosis using deep transfer learning. IEEE Trans Ind Inf 15(4):2446–2455
Shen C, Yang J, Xu J (2022) On federated learning with energy harvesting clients. In: ICASSP 2022–2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 8657–8661. IEEE
Short M, Twiddle J (2019) An industrial digitalization platform for condition monitoring and predictive maintenance of pumping equipment. Sensors 19(17):3781
Signoretti G, Silva M, Andrade P, Silva I, Sisinni E, Ferrari P (2021) An evolving tinyML compression algorithm for IoT environments based on data eccentricity. Sensors 21(12):4153
Sriram G (2022) Edge computing vs. cloud computing: an overview of big data challenges and opportunities for large enterprises. Int Res J Modern Eng Technol Sci 4(1):1331–1337
Stolojescu-Crisan C, Crisan C, Butunoi B-P (2021) An IoT-based smart home automation system. Sensors 21(11):3784
Taylor B, Marco VS, Wolff W, Elkhatib Y, Wang Z (2018) Adaptive deep learning model selection on embedded systems. ACM SIGPLAN Notices 53(6):31–43
Tham C-K, Sharma N (2021). In: Mukherjee A, De D, Ghosh SK, Buyya R (eds) Prescriptive maintenance using Markov decision process and GPU-accelerated edge computing. Springer, Cham, pp 167–181
Theissler A, Pérez-Velázquez J, Kettelgerdes M, Elger G (2021) Predictive maintenance enabled by machine learning: Use cases and challenges in the automotive industry. Reliab Eng Syst Saf 215:107864
Ucar A, Karakose M, Kırımça N (2024) Artificial intelligence for predictive maintenance applications: key components, trustworthiness, and future trends. Appl Sci 14(2):898
Van Der Westhuizen J, Lasenby J (2018) The unreasonable effectiveness of the forget gate. arXiv preprint arXiv:1804.04849
van Staden HE, Boute RN (2021) The effect of multi-sensor data on condition-based maintenance policies. Eur J Oper Res 290(2):585–600
Wan L, Chen Y, Li H, Li C (2020) Rolling-element bearing fault diagnosis using improved LeNet-5 network. Sensors 20(6):1693
Wang N, Ren S, Liu Y, Yang M, Wang J, Huisingh D (2020) An active preventive maintenance approach of complex equipment based on a novel product-service system operation mode. J Clean Prod 277:123365
Wang Y, Yan J, Yang Z, Dai Y, Wang J, Geng Y (2022) A novel federated transfer learning framework for intelligent diagnosis of insulation defects in gas-insulated switchgear. IEEE Trans Instrum Meas 71:1–11
Wang R, Yan F, Yu L, Shen C, Hu X, Chen J (2023) A federated transfer learning method with low-quality knowledge filtering and dynamic model aggregation for rolling bearing fault diagnosis. Mech Syst Signal Process 198:110413
Wen Y, Fashiar Rahman M, Xu H, Tseng T-LB (2022) Recent advances and trends of predictive maintenance from data-driven machine prognostics perspective. Measurement 187:110276
Wong SY, Ye X, Guo F, Goh HH (2022) Computational intelligence for preventive maintenance of power transformers. Appl Soft Comput 114:108129
Wu D, Liu S, Zhang L, Terpenny J, Gao RX, Kurfess T, Guzzo JA (2017) A fog computing-based framework for process monitoring and prognosis in cyber-manufacturing. J Manuf Syst 43:25–34
Xu D, Li Q, Zhu H (2019) Energy-saving computation offloading by joint data compression and resource allocation for mobile-edge computing. IEEE Commun Lett 23(4):704–707
Xu J, Liang Z, Li Y-F, Wang K (2021) Generalized condition-based maintenance optimization for multi-component systems considering stochastic dependency and imperfect maintenance. Reliab Eng Syst Saf 211:107592
Yang Q, Liu Y, Cheng Y, Kang Y, Chen T, Yu H (2019) Federated learning. Synth Lect Artif Intell Mach Learn 13(3):1–207
Yang L, Ye Z-S, Lee C-G, Yang S-F, Peng R (2019) A two-phase preventive maintenance policy considering imperfect repair and postponed replacement. Eur J Oper Res 274(3):966–977
Yang W, Chen J, Chen Z, Liao Y, Li W (2021) Federated transfer learning for bearing fault diagnosis based on averaging shared layers. In: 2021 Global Reliability and Prognostics and Health Management (PHM-Nanjing), pp. 1–7. IEEE

Yang W, Xiang W, Yang Y, Cheng P (2022) Optimizing federated learning with deep reinforcement learning for digital twin empowered industrial IoT. IEEE Trans Ind Inf 99:1
Yousefpour A, Fung C, Nguyen T, Kadiyala K, Jalali F, Niakanlahiji A, Kong J, Jue JP (2019) All one needs to know about fog computing and related edge computing paradigms: a complete survey. J Syst Architect 98:289–330
Yu W, Liu Y, Dillon TS, Rahayu W (2022) Edge computing-assisted IoT framework with an autoencoder for fault detection in manufacturing predictive maintenance. IEEE Trans Ind Inf 19:5701
Yu Y, Guo L, Gao H, He Y, You Z, Duan A (2023) FedCAE: a new federated learning framework for edge-cloud collaboration based machine fault diagnosis. IEEE Trans Ind Electron 71:4108
Zhang C, Ji W (2020) Edge computing enabled production anomalies detection and energy-efficient production decision approach for discrete manufacturing workshops. IEEE Access 8:158197–158207
Zhang W, Li X (2021) Federated transfer learning for intelligent fault diagnostics using deep adversarial networks with data privacy. IEEE/ASME Trans Mechatron 27(1):430–439
Zhang W, Li X (2022) Data privacy preserving federated transfer learning in machinery fault diagnostics using prior distributions. Struct Health Monit 21(4):1329–1344
Zhang W, Yang D, Wang H (2019) Data-driven methods for predictive maintenance of industrial equipment: a survey. IEEE Syst J 13(3):2213–2227
Zhang W, Lu Q, Yu Q, Li Z, Liu Y, Lo SK, Chen S, Xu X, Zhu L (2020) Blockchain-based federated learning for device failure detection in industrial IoT. IEEE Internet Things J 8(7):5926–5937
Zhang S, He C, Miao Z (2020) Intelligent fault diagnosis system based on vibration signal edge computing. In: 2020 Global Reliability and Prognostics and Health Management (PHM-Shanghai), pp. 1–5. IEEE
Zheng S, Ristovski K, Farahat A, Gupta C (2017) Long short-term memory network for remaining useful life estimation. In: 2017 IEEE International Conference on Prognostics and Health Management (ICPHM), pp. 88–95 (2017). IEEE
Zhu G, Liu D, Du Y, You C, Zhang J, Huang K (2020) Toward an intelligent edge: Wireless communication meets machine learning. IEEE Commun Mag 58(1):19–25
Zonta T, Da Costa CA, da Rosa Righi R, de Lima MJ, da Trindade ES, Li GP (2020) Predictive maintenance in the industry 4.0: a systematic literature review. Comput Ind Eng 150:106889

Authors and Affiliations

Abubakar Bala[1] · Rahimi Zaman Jusoh A. Rashid[2] · Idris Ismail[3] · Diego Oliva[4] · Noryanti Muhammad[5] · Sadiq M. Sait[6] · Khaled A. Al-Utaibi[7] · Temitope Ibrahim Amosa[3] · Kamran Ali Memon[1]

✉ Abubakar Bala
abala.ele@buk.edu.ng

Rahimi Zaman Jusoh A. Rashid
rahimi_zaman@petronas.com

Idris Ismail
idrism@utp.edu.my

Diego Oliva
diego.oliva@cucei.udg.mx

Noryanti Muhammad
noryanti@ump.edu.my

Sadiq M. Sait
sadiq@kfupm.edu.sa

Khaled A. Al-Utaibi
khaled.alutaibi.1@aramco.com

Temitope Ibrahim Amosa
amosatemitopeibrahim@gmail.com

Kamran Ali Memon
ali.kamran@kfupm.edu.sa

[1] Interdisciplinary Research Center for Communication Systems and Sensing (IRC-CSS), King Fahd University of Petroleum & Minerals, Dhahran 31261, Eastern Province, Saudi Arabia

[2] Project Delivery and Technology Department (PD &T), PETRONAS, Tower 1, PETRONAS Twin Towers 50088, Kuala Lumpur, Malaysia

[3] Electrical and Electronics Engineering Department, Universiti Teknologi PETRONAS, Seri Iskandar 32610, Perak, Malaysia

[4] Depto. de Ingeniería Electro-Fotónica, Universidad de Guadalajara, CUCEI, Guadalajara 44430, Jalisco, Mexico

[5] Centre for Mathematical Sciences, Centre of Excellence for Artificial Intelligence & Data Science, Universiti Malaysia Pahang, Lebuh Persiaran Tun Khalil Yaakob, Kuantan 26300, Pahang, Malaysia

[6] Department of Computer Engineering, King Fahd University of Petroleum and Minerals, Dhahran 31261, Eastern Province, Saudi Arabia

[7] Corporate Digital Factory Department, Saudi Aramco, Dhahran 31311, Eastern Province, Saudi Arabia