Task offloading in Cloud-Edge Collaboration-based Cyber Physical Machine Tool

Chuting Wang[a,b], Ruifeng Guo[b],Haoyu Yu[a,b], Yi Hu[b], Chao Liu [c], Changyi Deng[d*]

[a] *University of Chinese Academy of Sciences, Beijing 100049, PR China*

[b] *Shenyang Institute of Computing Technology Chinese Academy of Sciences, Shenyang 110168, PR China*

[c] *College of Engineering and Physical Sciences, Aston University, Birmingham B47ET, UK*

[d] *Institute of Software , China Industrial Control Systems Cyber Emergency Response Team, Beijing 100040, PR China*

ARTICLE INFO

Keywords:
Cyber-Physical Machine Tool
Digital twin
Cloud-edge collaboration
Task offloading

ABSTRACT

The Cyber-Physical Machine Tool (CPMT) is a promising solution for the next generation of machine tool digitalization and servitization due to its excellent interconnection, intelligence, adaptability, and autonomy. The rapid development of next-generation information technologies, such as the Internet of Things (IoT) and artificial intelligence (AI), provided richer services for CPMT but also led to problems of idle on-site computing resources, and excessive pressure on the cloud, slow service response and poor privacy. To solve the above problems, this paper proposes a cloud-edge collaboration-based CPMT architecture, which makes full use of the computing resources of existing devices in the industrial sites, offloads digital twin (DT) modeling and data processing from the cloud to the edge, and provides microservice interfaces for users at the edge. Given the limited computing resources available in the field and the demand for latency-sensitive applications, task offloading methods aimed at response speed and load balancing are proposed, respectively. Finally, a case of machine tool Prognostics and Health Management (PHM) service is presented, in which the proposed method is used to perform tool wear monitoring, prediction, and health management.

1 Introduction

Machine tools have always played a decisive role in the field of manufacturing [1]. Their performance directly affects the quality and efficiency of the products. [2]. With the deep integration of a new generation of information technology (IT) and advanced manufacturing technology, machine tools have gradually evolved from the initial Machine Tool 1.0, which is machine-driven but human-operated, into the current Machine Tool 3.0, which is computer numerically controlled, on the way to the era of Machine Tool 4.0. The rapid development of intelligent manufacturing and Industry 4.0 demand Machine Tool 4.0 to be more intelligent, well interconnected, more flexible, and more autonomous [3]. As a typical representative of Machine Tool 4.0, Cyber-Physical Machine Tool (CPMT) [4] integrates the machine tools and manufacturing processes with computation and networking based on the digital twin (DT) technology, which aims to transform machine tools from physical products to product-service systems and cloud resources.

As the kernel of CPMT, DT digitally creates a machine mirror for physical machine tools to provide real-time supervision and feedback control on the machine tool status and machining process, enabling the integration of the physical world with the cyber world. Most of the traditional architectures of DT are based on "cloud-end" [5,6]. In other words, the device is only responsible for performing production tasks and collecting real-time data, while all data processing tasks are uploaded to the cloud. However, with the arrival of the Internet of Everything (IoE) era and the rapid evolution of intelligent algorithms, the scale of data and computation in industrial sites have increased dramatically. Cloud computing may fail to process various and massive sensed data in a timely manner, resulting in intolerable delays [7,8] and inability to meet the demand for real-time data interaction [9], eventually causing the isolation of physical machine tools from their mirrors in the DT machines. In addition, the construction and application of cloud platforms in the industrial field have lagged far behind their theoretical development [10]. Moreover, numerous edge devices with computing capabilities have already been deployed into industrial sites before the emergence of cloud computing.

Considering the aim of the existing equipment assets in industrial sites and the excellent performance of edge computing is to provide low latency, high bandwidth, and mass accessible devices [11–13], this paper utilizes edge computing to compensate for the deficiencies of cloud computing and proposes a cloud-edge collaboration-based CPMT, which further enhances the digitalization and servitization of machine tools. The main contributions of this paper are as follows.

- An architecture of cloud edge collaboration-based CPMT is proposed.
- Explored task offloading technique to improve responsiveness.
- A task offloading algorithm is proposed to provide load balancing while maintaining system real-time and throughput rate requirements, ensuring CPMT scalability and supporting horizontal integration of machine tools.
- An application service case of cloud-edge collaboration based CPMT based on cloud-edge collaboration is given to validate the feasibility and advantages.

The rest of this paper is arranged as follows. Section 2 reviews the development of CPMT and the latest works on cloud edge collaboration in academia and industry. Section 3 profiles the architecture of the cloud-edge collaboration-based CPMT. Section 4 depicts two specific algorithms for deploying CPMT through cloud-edge collaboration. Section 5 presents an application case. Section 6 concludes this paper and offers suggestions for further research.

2 Related works

Machine tools are involved in almost all manufacturing processes and have occupied an iconic position in the manufacturing industry since their introduction. To a large extent, the development of machine tool technology mirrored that of the industrial revolution. With the advent of the new era of Industry 4.0, the digitalization and servitization of machine tools are becoming an emerging research trend [14]. This section reviews the evolution of CPMT, provides a comprehensive overview of the work and technologies related to cloud-edge collaboration, and further identifies the research gaps.

Xu [3] reviewed the history of machine tools, pointed out that the development of machine tools should come to the era of Machine Tool 4.0, innovatively put forward the conceptual model of CPMT, and briefly introduced the critical components and functions of

CPMT, which the total vertical and horizontal integration of machine tools can be achieved. On this basis, Liu et al. [4] clarified the definition of CPMT, stating that CPMT combines a machine tool, machining processes, computing, and a network in which embedded computing monitors and controls the machining process via feedback loops. The machining process can influence computing and vice versa. Moreover, a generic system architecture for CPMT was also suggested [2], which looked at the development techniques for the DT of machine tool (MTDT), the kernel of CPMT, in detail, providing directions for converting current CNC machine tools to CPMT. Deng et al. [15] deployed a wireless sensor network (WSN) on an open CNC system and extended it to CPMT, enabling real-time supervision and control of the processing. To solve the problem of interacting with heterogeneous data from multiple sources, Liu et al. [16] proposed an MTConnect-based CPMT architecture, including Physical devices, Networks, machine tool cyber twin and cloud. Moreover, a prototype system was designed based on the proposed architecture to deliver unified and efficient data communication, fusion, and management. Zhu et al. [17] proposed a user-centered CPMT information provision model that divides CPMT into Physical machine tools and processes, MTDT, Cloud-based services, and Human-Machine Interfaces (HMIs). It provided and distributed pertinent information and data to the correct users in the proper scenarios and at the right time and further increased the efficiency and productivity of CPMT.

Cloud computing has plenteous computational and storage resources for complex big data analysis, but its real-time behavior is poor. Edge computing reduces the pressure on the cloud by concentrating computation and services close to the data source. Therefore, many algorithms have been proposed to offload the task to improve performance or reduce energy consumption. Cao et al. [18] proposed a multi-agent deep reinforcement learning scheme that enables edge devices to work together and significantly reduce computational latency. Liu et al. [19] proposed a holistic offloading method that minimizes the latency by determining whether to offload the tasks waiting in the buffer through a one-dimensional search algorithm. Joshi et al. [20] proposed an energy-efficient task offloading with delay awareness, trading off between energy consumption and latency in communication. Tang et al. [21] performed dynamic resource allocation between the edge and the cloud, using an optimal algorithm to transfer data from the cloud to the edge and assign resources to them. Qi et al. [22] analyzed the different requirements and uses of data generated at the unit-level, system-level, and complex system-level at the time scale for CPS and DT. They then proposed ideal solutions for intelligent manufacturing, which utilize edge computing, fog computing, and cloud computing at different-level CPS and DT. Yang et al. [23] proposed an open, scalable architecture for intelligent cloud manufacturing systems with collaborative edge and cloud computing to support the real-time response of latency-sensitive applications by deploying hierarchical edge gateways. Zhang et al. [24] focus on the computation efficiency of the twin established, developed a cyber-physical machine tool based on edge computing techniques, which detect anomalies in edge data in terms of both monadic outliers in the edge data itself and multivariate parameter correlations between edge units, shortening the mapping latency and reducing the workload in the cloud.

Current research shows that academia and industry strive to implement the digitalization and servitization of machine tools. However, most previous research focuses on creating DT models of machine tools, with less research on DT in the machining process. Meanwhile, most of the specific services provided by CPMT rely on the cloud. Nonetheless, this cloud-centric data processing in industrial sites may cause information lags due to bandwidth and transmission delays, resulting in information desynchronization between physical space and cyberspace. Relying on cloud computing alone to implement CPMT is too idealistic. At present, edge computing technology is quickly evolving, and the cloud-edge cooperation technique may achieve the synergistic coupling between edge computing and cloud computing and unlock the value of data [25], which is widely applied in various industries. However, in the application of CPMT, edge computing is only used at the data fusion stage [24], which does not maximize the utility of cloud-edge collaboration. This paper innovatively proposes a CPMT architecture based on cloud-edge collaboration to address the above problems. It provides new ideas and ways for the servitization transformation of existing machine tools by exploiting the complementary properties of edge computing and cloud computing.

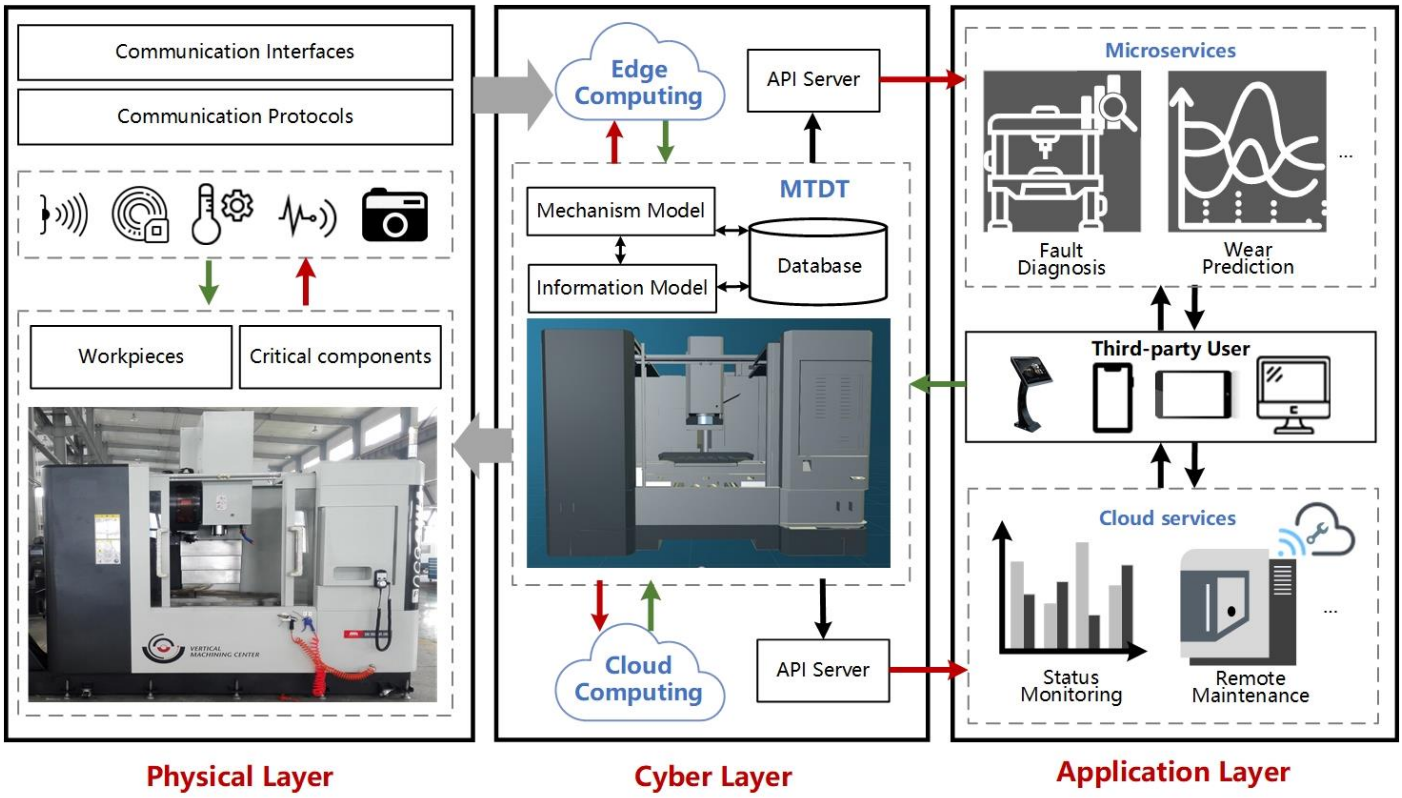

Fig. 1 Architecture of Cloud-edge collaboration-based CPMT

3 Cloud-edge collaboration-based CPMT

3.1 Overall architecture

This section provides an overall architecture of cloud-edge collaboration-based CPMT. As shown in Fig. 1, the overall architecture consists of three layers: physical layer, cyber layer, and application layer. In the figure, the red arrows represent data transfer, while the green arrows represent result feedback. The physical layer covers all the physical items of CPMT. The cyber layer is the crux of this architecture, where the physical CNC machine tool is virtualized as the corresponding MTDT through cloud computing and edge computing. By describing CNC machine tools' multidimensional and multiscale characteristics, MTDT, driven by real-time data, can accurately depict machine tools' actual behavior and operating state. MTDT infers the future operating state and trends of the behavioral characteristics of machine tools and machining processes via iterative model and data operations. Moreover, use them to support essential services such as prediction, evaluation, and optimization of machine tools and machining processes [26,27]. MTDT modeling can be divided into two parts [28,29]: visual 3D modeling and semantic modeling. Fig.1 shows the 3D visualization model we built for a typical machine tool VMC850 in our previous work. The semantic model [29,30] indicates structured management of collected data and analysis processing. The cyber layer performs feedback control of the physical layer through MTDT and provides an external service interface via APIs to support the servitization of the machine tool. The final layer is the application layer, which provides cloud services and microservices to third-party users.

The most vital distinction between the proposed architecture and traditional architectures of CPMT depends on its focus on cloud-edge collaboration. Compared to traditional cloud-based CPMT [2,15,16], the cloud-edge collaborative mode can fully utilize the computing resources of existing equipment in the workshop, thus maximizing the value of edge computing and the cloud [31]. Cloud-centric servers have an immense size, with high computing power, reliability, and scalability [32,33]. However, with the rapid development of IoE, users are more demanding on service responsiveness and security. Although cloud computing can provide an efficient computing platform for processing big data, the network bandwidth growth rate at this stage can no longer match the explosive growth of data [34]. The cost of network bandwidth is also decreases slower than the cost of hardware resources such as CPU and memory. A single cloud-based computing mode cannot satisfy users' demands for real-time security and low power consumption [35]. As a complement and extension of cloud computing, edge computing makes

full use of idle device resources and harnesses the computing power of endpoints at the network edge. It relieves pressure on network bandwidth and data centers by absorbing part or whole of the computing tasks, solves the problem of data transmission latency, improves service response time, and enhances data security by limiting the exploitation of private data inside the firewall.

Cloud-based architectures can provide non-real-time services such as condition monitoring, process planning, and remote maintenance, while edge computing-based architectures enable real-time services such as error compensation, surface roughness prediction, and anomaly detection [14]. In this paper, we adopt the cloud-edge collaboration approach to drive CPMT in order to enhance the serviceability of machine tools. More details about the cloud-edge collaboration-based CPMT are explicated in the section below.

3.2 Physical devices

This paper extends the cloud-based CPMT by deploying a series of edge devices. Fig. 2 presents the physical devices of the CPMT based on cloud-edge collaboration. The end devices contain CNC machine components and data acquisition devices. Components such as CNC controllers provide internal data about the machine, such as spindle speed, axis position and velocity, program information, and system information. Data acquisition devices, including sensors for vibration, temperature, and acoustic emission (AE), access external data to the CPMT.

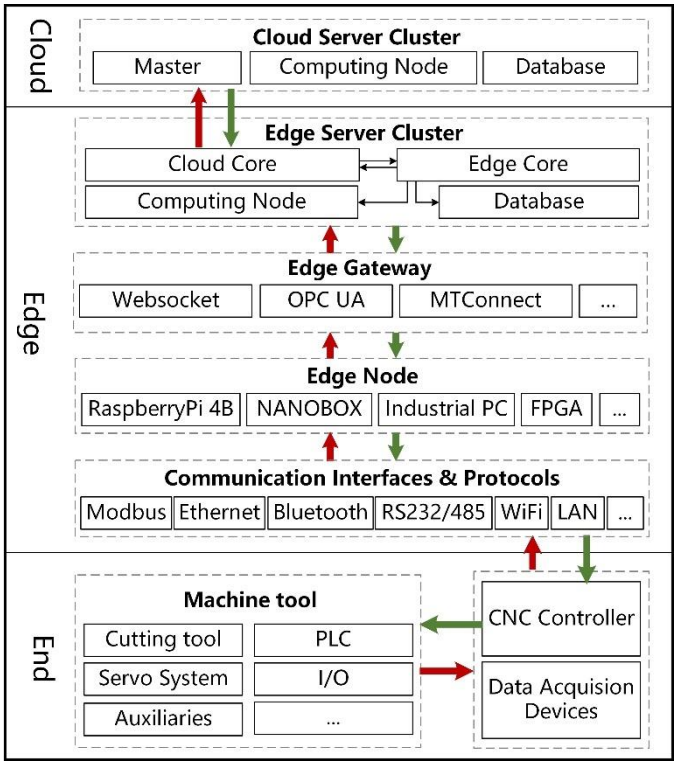

Fig. 2 Physical devices of the CPMT

The real-time data collected during manufacturing is sent to the edge and cloud for further analysis through various communication protocols and interfaces. The cloud and edge devices carry the kernel of CPMT, the MTDT, which is the basis for the intelligence and servitization of CNC machines. As shown in Fig. 2, the edge nodes are composed of devices with specific computing capabilities such as Raspberry Pi, NANOBOX, IPCs, and FPGAs. In order to efficiently gather heterogeneous manufacturing data and transfer it to the edge server or cloud for resource virtualization and decision analysis, the edge gateway supports standardized data communication methods such as MTConnect and OPC UA to eliminate data heterogeneity [9,36,37] and enable efficient data interaction. The edge server cluster contains numerous servers serving different purposes, including Cloud Core and Edge Core, in addition to the basic compute and storage modules. Cloud Core reports the status of edge resources to the Master node in the cloud and issues cloud commands to Edge Core, while Edge Core manages the edge resources. The Edge nodes collaborate with the Edge Server Cluster to process data from the end devices and upload essential and valuable product information to the cloud for storage and analysis.

The cloud server cluster consolidates, stores, and shares the data. The Master, as the control node, is responsible for resource scheduling, management, and control of the entire cluster.

3.3 Cyber layer

The Cyber layer establishes and deploys the MTDT through cloud-edge collaboration. In contrast to traditional CNC machine tools, the MTDT of CPMT achieves a profound integration of the model and data by constructing a digital mapping of the machine tool. MTDT reflects the geometric and physical characteristics, behavioral coupling relationships, and evolutionary rules of CNC machine tools and machining processes [36] and forms an integrated decision that feeds back to guide machining toward closed-loop optimization. Based on the five-dimensional model theory of the digital twin [38–40], we divide MTDT into three main modules: the information model, the mechanism model, and the database. Among them, the information model reflects the coupling between components and systems by portraying the logical structure of the machine tool and provides standardized and scalable data management [41]. The mechanism model is built by applying artificial intelligence (AI) algorithms to data processing and combining them with expert experience and domain knowledge to jointly build a comprehensive high-fidelity and high-accuracy model that can reflect the behavior and rules of physical entities in order to transform a large amount of raw machine tool data into meaningful information to support machine tool service. Databases provide secure, reliable, and adequate storage and management of real-time data and valuable historical information generated by machining.

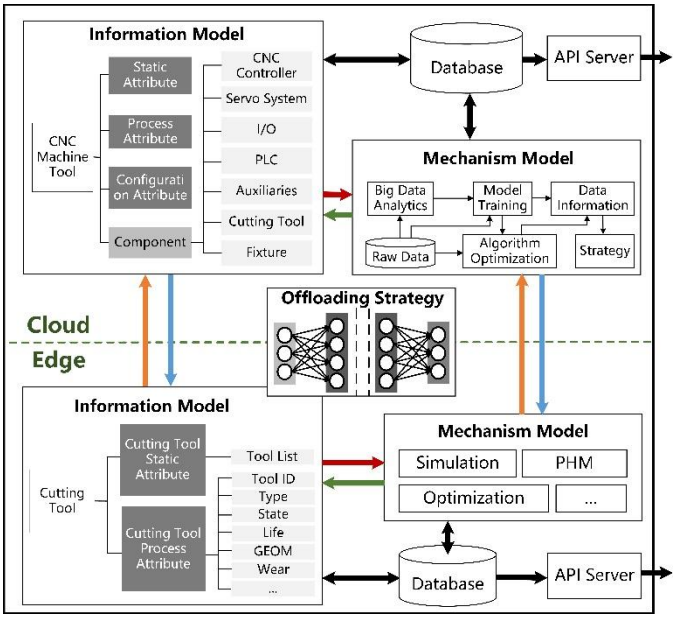

Fig. 3 Cyber layer

CNC machine tools involve multiple subsystems, such as electrical and mechanical, among which there are coupling relationships [42]. In our previous work [37,43], a generic information model for machine tools was built by analyzing the hierarchical structure of CNC machine tools. Considering the composability of DT models [36], the edge side can flexibly extract and combine models without building a complete MTDT. This paper provides a reference model of MTDT driven by cloud and edge computing in collaboration, as shown in Fig. 3. A complete machine tool information model is formed in the cloud, integrating historical data and valuable information to build an expert knowledge base of CNC machine tools. Cloud computing has more substantial arithmetic power to optimize algorithms by training models and formulate task offloading strategies. The edge builds or downloads DT models according to the service requirements and allocates them to different edge devices respectively to realize the application of DT in different stages and scenarios. Ultimately, the related cloud services or microservices are provided by API. The collaborative cloud-edge approach improves the efficiency of DT modeling and lowers the application threshold simultaneously.

In Fig. 3, the red arrows indicate the data transfer, and the green arrows indicate result feedback. The mechanism model, consisting of algorithms for simulation, optimization, Prognostics and Health Management (PHM), obtains specific standardized data directly from the information model for analysis and processing and feeds the results back to the information model, which issues commands to the CNC controller to automatically adjust machining or provide adequate decision support to the operator. CNC machine characteristics are not static during machining. To ensure the reliability/faithfulness of the model, the MTDT is continuously updated. In Fig. 3, the orange arrows are used to indicate model updates while the blue arrows are used to indicate model offloading. The cloud trains the model with time-stamped historical data from the edge and offloads the optimized model to the edge. The edge processes real-time data from end devices to improve service responsiveness. In this layer, we allocate computational tasks by developing an offloading strategy, the details of which are discussed in Section 4.

3.4 Application layer

The application layer is the final layer of the cloud-edge collaboration CPMT architecture. After processing the data in the Cyber layer, various cloud and edge computing-driven services can be implemented to meet

the needs of process optimization and production quality control, such as PHM, 3D visualization, and condition monitoring. The application layer also provides various types of HMIs to facilitate the interaction of different stakeholders (e.g., machine operators, maintenance personnel, shop floor managers, machine tool manufacturers) with the CNC machine to further promote the digitalization and service-orientation of the machine.

4 Methodologies of cloud edge collaboration for CPMT

CPMT collects data and analyzes it through MTDT. Most existing CPMT models [5] transfer the collected data directly to the cloud center for storage and analysis. However, in CPMT deployments, numerous computing resources of edge devices on the shop floor are idle. Therefore, the centralized data processing model adopted in the CPMT conceptual model needs to be adapted accordingly by migrating computational tasks with high real-time requirements to the edge. Thus, while relieving network bandwidth pressure and reducing the load on cloud datacenters, the existing devices are fully utilized to provide more convenient computing resources for industrial field data analysis.

In this section, we first establish a system model for the task offloading problem of CPMT and then present two task offloading algorithms for task response speed and load balancing requirements, respectively.

4.1 System model

The primary purpose of this work is to minimize service delay under the proposed framework. Service delay [44] is the time required to complete a task request. The edge is sited between the end devices and the cloud, which contains massive devices with computing power and can handle most service requests to reduce the overall service delay.

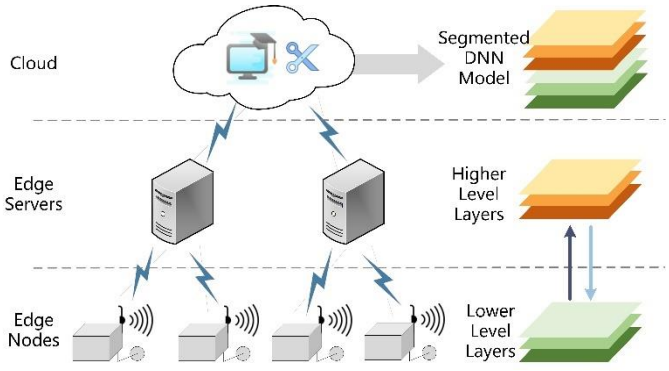

Fig. 4 Task processing flow

Fig. 4 illustrates the task processing flow for the proposed framework work. This paper aims to process latency-sensitive tasks at the edge as much as possible. That means the cloud only trains models and handles the tasks that cannot be handled at the edge. Most of the real-time services provided by CPMT, such as tool wear monitoring and fault diagnosis, rely on machine learning algorithms represented by Deep Neural Networks (DNN). Therefore, this paper assumes that all tasks are machine learning tasks that pre-trained DNN model processes in the cloud, and the output is obtained by performing inference. After data capture, the tasks are sliced, and the corresponding container pipeline is created for each model slice and evaluated. Finally, the tasks are distributed between the edge nodes and the edge servers according to the offloading requirements.

1) Network model

The normalized network model in this paper is defined as follows: the set of edge nodes is $\mathbb{N}$, $\mathbb{N} = \{N_i | \mathrm{i} = 1,2,\dots,\mathrm{N}\}$, $|\mathbb{N}| = \mathrm{N}$. Any node $N_i \in \mathbb{N}$ holds a task R. If a single node holds multiple tasks, the node can be split into multiple virtual nodes, and each node holds only one task. Tasks can be executed locally (edge node) or offloaded to an edge server, and the results are transmitted back to the edge node. It is assumed that the selected server contains the resources required for task execution.

2) Task model

The task is sliced according to the neural network structure, and the task R is split into n subtasks, $R = \{R_i | i = 1,2,\dots,n\}$. D_i denotes the size of the output data after the execution of subtask R_i. $l_R$denotes the deadline of task R. Each task R can be left local (edge node), offloaded to the server as a whole, or partially offloaded to the edge server for execution. The task segment point is denoted as j, which is located after the jth subtask has been executed. For different types of neural network layers, different layer configurations can lead to significant variations in latency. Kang [45] et al. varied the configurable parameters of various neural network layers and measured the latency of each configuration to construct latency prediction models for each type of layer so that the latency of the DNN constituent layers could be estimated without executing the DNN. Based on this, the latency of a node executing subtask R_i is denoted as $f_{node}(R_i)$and the latency of a server executing subtask R_i is denoted as $f_{server}(R_i)$,where $i = 1,2,\dots,n$.

The service delay T_R of task R consists of node execution latency, server execution latency and transmission delay. Where the edge node task execution delay is:

$$t_{node} = \sum_{i=0}^{j} f_{node}(R_i) \quad (1)$$

The edge server execution delay is:

$$t_{server} = \sum_{i=j+1}^{n} f_{server}(R_i) \quad (2)$$

The data is transmitted between the node and the server and the transmission delay can be calculated according to the *Shannon formula* as:

$$t_{tran} = D_i / \mathrm{B} \log 2(1 + p_i |h_i|^2 / \sigma^2) \quad (3)$$

B is the available bandwidth between the node and the server, p_i is the transmit power of node i, h_i is the channel gain, and σ^2 is the white Gaussian noise power inside the channel.

In order to make full use of edge node resources and reduce data discard, this paper processes the task through Multiple Instruction stream Single Data stream (MISD) pipeline structure.

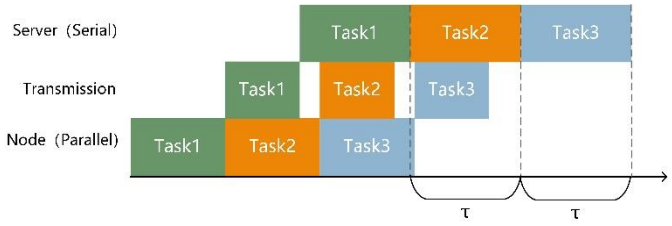

Fig. 5 MISD pipeline structure

As shown in Fig. 5, the nodes execute in parallel, and the server executes serially by non-preemptive CPU allocation. Every timeslice τ elapses, the system outputs a set of processing results,$\tau = \max\{t_{node}, t_{tran}, t_{server}\}$. When the pipeline is stable and fully loaded, the highest speed of data processing results can be guaranteed within each timeslice [46]. At this point, the system throughput rate $\theta = 1/\tau$, and a higher throughput rate means fewer data is discarded.

4.2 Offloading strategy

The offloading segment point of a DNN model depends on its topology, which is reflected in the variation in computation and data size at each layer. In addition, dynamic factors such as network state and device load can also affect the choice of offloading segment points. We propose an approach that intelligently slices tasks and formulates offloading strategies. It includes two phases, static configuration, and service execution. In the static configuration phase, the edge nodes and servers are configured to obtain the delay prediction model of the DNN layer spectrum from the cloud database and store it on the corresponding nodes and servers. In the service execution phase, the system analyzes the DNN layer types, extracts its configurations, and uses the stored prediction models to value the delay of each layer's execution delay on the nodes and servers. Furthermore, an offloading strategy for specific optimization goals (response speed/load balance) is formulated by combining the current network conditions. Finally, the DNN is executed by distributing tasks across nodes and servers according to the offload policy developed.

4.2.1 Offloading strategy1: Fastest Response

Task response speed is an essential indicator of service quality and a vital optimization goal for edge computing task offloading, and **Algorithm 1** develops an offloading strategy to improve the response speed. Fig. 6 shows three possible scenarios when a new task arrives at the server: a) the server is busy, and the new task needs to wait for the previous task to complete; b) the new task arrives without waiting for the server, but the server incurs idle time; c) the server happens to be idle when the task arrives. Note that the server execution delay of previous tasks is mt_{server}, then service delay of task R, $T_R = \max\{t_{node} + t_{tran},\ mt_{server}\} + t_{server}$.The detailed offloading process is as follows.

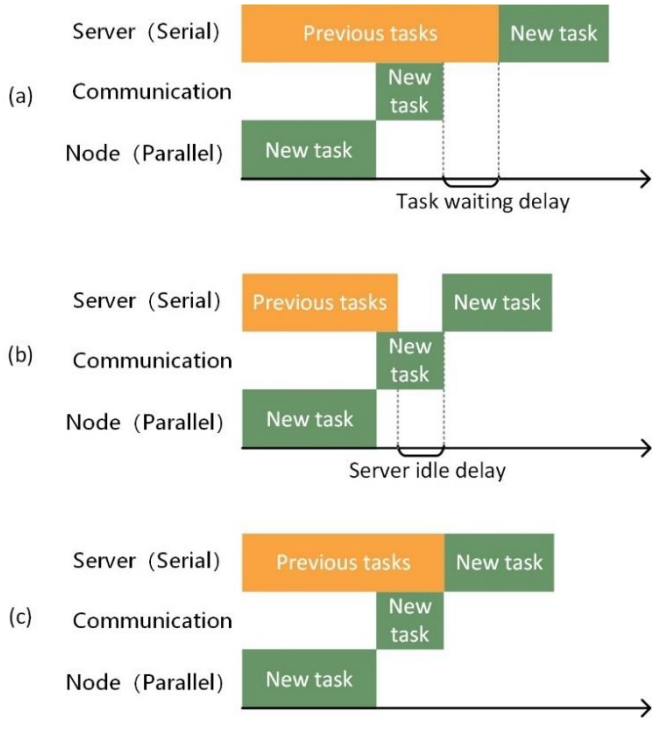

Fig. 6 Three possible scenarios when a new task arrives at the server

Algorithm 1 Offloading Strategy: Fastest Response

1: **Input:**
2: N： number of the subtasks
3: $\{R_i | i = 1, \ldots, N\}$： the i-th subtask
4: D_i: output data size of R_i
5: $f(S_i)$: latency of executing R_i
6: l: deadline of task R

7: B: current available bandwidth
8: m: current number of nodes that connect to the server
9: **Output:**
10: Offloading Strategy: the offloading segment point J_{best}
11: **procedure**
12: **for each** i in 1,…,N **do**
13: $TN_i \leftarrow f_{node}(R_i)$;
14: $TS_i \leftarrow f_{server}(R_i)$;
15: **end for**
16: Let $T_R \leftarrow l, J_{best} \leftarrow null$;
17: **for** $j = 0, j \leq N, j = j + 1$ **do**
18: $t_{node} = \sum_{i=0}^{j} TN_i$;
19: $t_{server} = \sum_{i=j+1}^{N} TS_i$;
20: $t_{tran} = D_i / \mathrm{B} \log 2(1 + p_i|h_i|^2/\sigma^2)$;
21: $T_j = \max\{t_{node} + t_{tran}, mt_{server}\} + t_{server}$;
22: **if** $T_j \leq T_R$ **then**
23: $T_R \leftarrow T_j$;
24: $J_{best} \leftarrow j$;
25: **end if**
26: **end for**
27: **if** $J_{best} = null$ **then**
28: Send task R to the *Cloud* for processing;
29: **end if**;
30: **return** J_{best}

When a new task R arrives, we slice it into N subtasks at the beginning and initialize the execution delay of each subtask at the node and server, denoted $TN_i$and TS_i, respectively (Line 12-15 in Alg. 1). Next, we try to slice the tasks at each candidate segment point and calculate the corresponding service delay T_j according to the model in Section 4.1 (lines 17-21 of Alg. 1). Finally, we compare the service delay corresponding to each candidate segment point and select the one with the shortest service delay as the output of the discharged segment point (Line 22-25 in Alg.1). Note that if the service delay corresponding to all candidate segment points exceeds the task delay, the edge cannot process the task R and send it to the cloud (Line 27-29 in Alg.1).

4.2.2 Offloading strategy2: Load Balance

As we know, the computational power of edge servers is much better than that of edge nodes, and in order to ensure the response speed of node tasks, **Algorithm 1** tends to migrate most of the tasks to the server. However, the number of edge nodes on the shop floor is often more significant than the number of edge servers. This offloading method may cause the servers to be overloaded, and there may be no server resources available when new tasks are accessed, which is not conducive to system scalability and will result in the waste of a large amount of node computing resources. In order to avoid the above puzzles, we propose a new task offloading algorithm to maximize the number of server access nodes based on the real-time performance and throughput rate guarantee of tasks. The specific algorithm is as follows.

Algorithm 2 Offloading Strategy: Load Balance

1: **Input:**
2: N: number of the subtasks
3: $\{R_i | i = 1, \ldots, N\}$: the i -th subtask
4: D_i: output data size of R_i
5: $f(R_i)$: delay of executing R_i
6: l: deadline of task R
7: B: current available bandwidth
8: θ: the threshold of throughput rate
9: **Output:**
10: Offloading Strategy: the offloading segment point J_{best}
11: M: the number of nodes that can connect to the server
12: **procedure**
13: **for each** i in 1,…,N **do**
14: $TN_i \leftarrow f_{node}(R_i)$;
15: $TS_i \leftarrow f_{server}(R_i)$;
16: **end for**
17: Let $J_{best} \leftarrow null$, P $\leftarrow \emptyset$
18: **for** $j = 0, j \leq N, j = j + 1$ **do**
19: $t_{node} = \sum_{i=0}^{j} TN_i$;
20: $t_{server} = \sum_{i=j+1}^{N} TS_i$;
21: **for** M_j=0 **do**
22: $\tau_j = \max\{t_{node}, t_{tran}, t_{server}\}$, $\theta_j = 1/\tau_j$;
23: $T_R = t_{node} + t_{tran} + M_j t_{server}$;
24: **if** $T_R \leq l$ **then**
25: $M_j = M_j + 1$;
26: **end if**
27: **end for**
28: **if** $\theta_j \geq \theta$ **then**
29: P $\leftarrow$ P $\cup \{p_j\}$, $p_j = (j, M_j, \theta_j)$;
30: **end if**
31: **end for**
32: Select the p_j with maximum M_j in P
33: $J_{best} \leftarrow j$
34: **return** J_{best};

Similar to **Algorithm 1**, **Algorithm 2** first initializes the execution delay of each subtask on the node and server (Line 13-16 in Alg.2) and calculates the execution delay of the node t_{node} and server t_{server} corresponding to each candidate segment point (Line 18-20 in Alg.2). The transmission delay is obtained from actual measurements, and the throughput rate and service delay are calculated according to the system model (Line21-23 in Alg.2). The algorithm calculates the maximum number of server accessible nodes M_j corresponding to a candidate segment point j in meeting the real-time requirement and stores the candidate segment points meeting the throughput rate requirement in the set P (Line 24-31 in Alg.2). Finally, the maximum value of M_j

is searched in the set P, and its corresponding segment point is returned as the offloading segment point (Line 32-34 in Alg.2).

5 Case Study

The CNC machine tool is typical of complex and sophisticated machining equipment, and its health directly impacts product quality and production stability. The operator's subjective experience is primarily relied upon in the traditional CNC machining process to determine the health of the machine tool and perform maintenance. However, the CNC machine tools are located in a complex environment with complex failure information and a high probability of failure. On the one hand, failure to troubleshoot or replace components on time will result in machine health deterioration, shortened life, reduced processing quality, and even significant loss of life and property. On the other hand, frequent periodic maintenance can result in decreased utilization, wasted resources, and uncertainty in an otherwise stable manufacturing system [47]. To address the above issues, the application layer of cloud edge collaboration-based CPMT provides users with PHM services for machine tools that combine historical data with machine learning algorithms to predict failures or maintenance events in advance, thereby significantly reducing downtime and maintenance costs. The PHM has become a major concern for academia and industry.

PHM relies on the fault alarm information provided by the CNC system to prompt the operator to resolve CNC machine obstacles early and quickly to ensure smooth production. However, in today's CNC equipment manufacturing industry, there is a wide variety of machine tools, and in addition to the machine faults defined by public standards, there are numerous fault messages customized by each machine tool manufacturer. This heterogeneous fault information will cause each CNC machine tool to become an information island, and it is difficult for the workshop or factory to realize the unified coding display, archiving, integration, and sharing of machine tool fault information. That brings great difficulties to the provision of PHM services and the hidden danger of equipment reliability for the horizontal integration of machine tools.

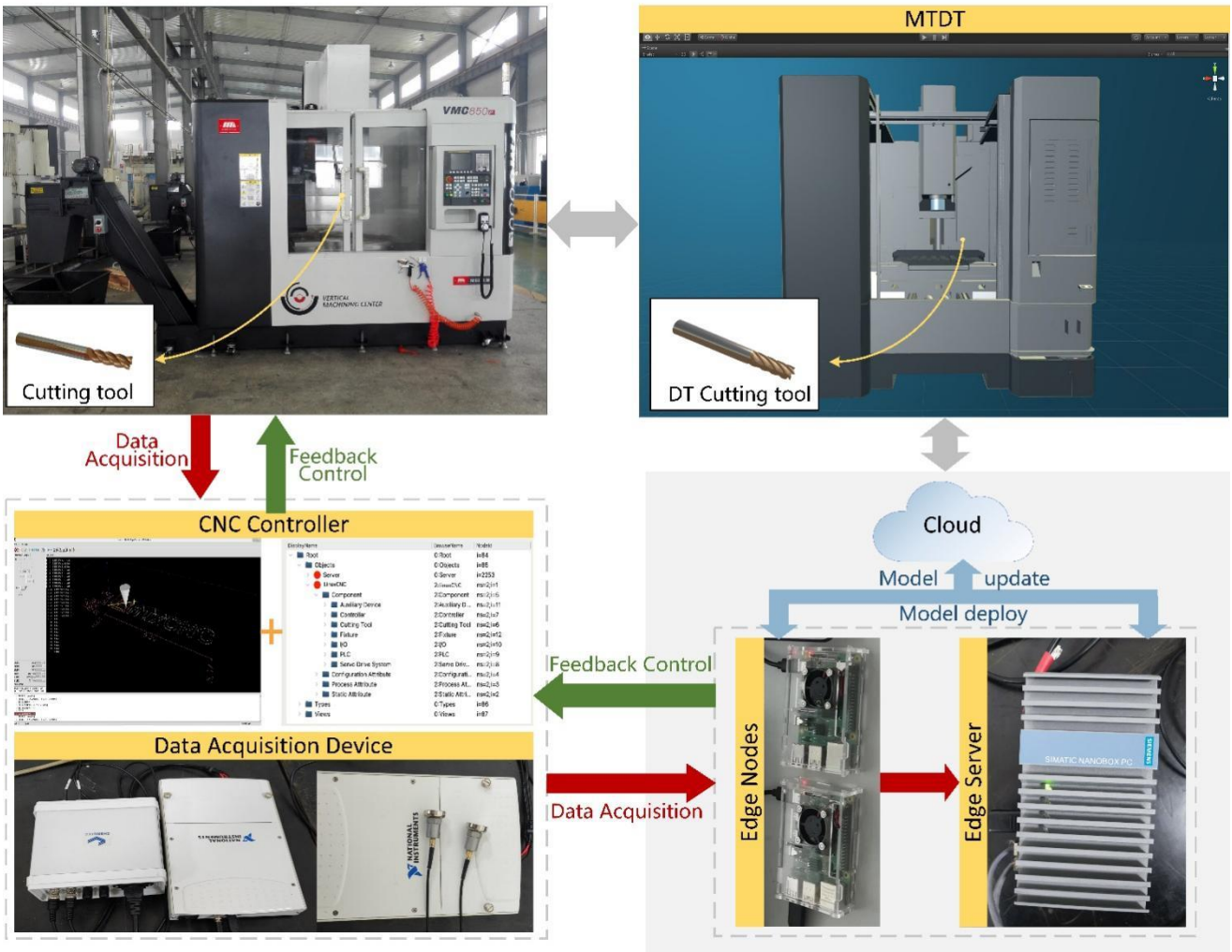

Fig. 7 Prototype system for cloud-edge collaboration-based CPMT

CPMT builds a DT model of machine tool fault based on the historical information of equipment, realizes a unified representation of CNC machine tool fault information, and provides failure diagnosis and predictive maintenance for machine tools by analyzing the changes in temperature, load, and position of the environment, machining object, and equipment of the machine tool by sensor acquisition and computer technology. This section takes the machine tool PHM service as an example to further demonstrate the feasibility and benefits of the proposed cloud edge collaboration-based CPMT, including the introduction of the physical devices and experimental environment, the establishment and deployment of the machine tool fault DT model in the cyber layer, and the cutting tool PHM service provided in the application layer.

5.1 Physical devices and experimental environment

Fig. 7 shows the prototype system for cloud-edge collaboration-based CPMT. An OPC UA server and a machine tool PHM application service are developed based on the LinuxCNC open-source platform and integrated into the CNC controller. This paper uses RaspberryPi 4B as the edge node (end) and SIMATIC NANOBOX PC as the edge server to apply in the cloud, edge, and end, respectively Kubernetes, KubeEdge, and Edge X Foundry platforms to build a cloud-edge collaborative environment. As shown in Fig. 7, the cloud collects the state information as a model training set to train the diagnostic and prediction models. The trained models are deployed to the edge nodes and servers. Real-time data is analyzed at the edge for machine tool fault diagnosis and prediction, and the PLC provides fault codes to the CNC system or master computer. The user indexes the fault code in the fault dictionary database through the CNC controller control panel and APIs of the cloud services or microservices to obtain fault information such as fault name, fault parameters, fault rank, fault reflection, and fault handling methods to provide decision support to the operator.

5.2 Cyber layer: MTDT modeling and deployment

5.2.1 Fault information data dictionary of CNC machine tool

This section establishes the MTDT model based on the standard "NC Equipment Fault Information Data Dictionary-CNC Machine Tools" (accepted, not yet released). First, analyze the fault components and hierarchy of CNC machine tool fault information, and make uniform fault classification and coding for CNC machine tools from different manufacturers. Then, a "CNC machine tool fault information data dictionary" is established to facilitate users to view the relevant fault information of CNC machine tools easily and quickly. The CNC machine tool fault information data dictionary contains two parts: machine tool fault code and fault information card. The fault code is the unique index of the machine's fault information, and the fault information card contains the fault code, fault name, fault rank, fault parameters, fault reflection, fault content, suggested treatment, and system continuation method.

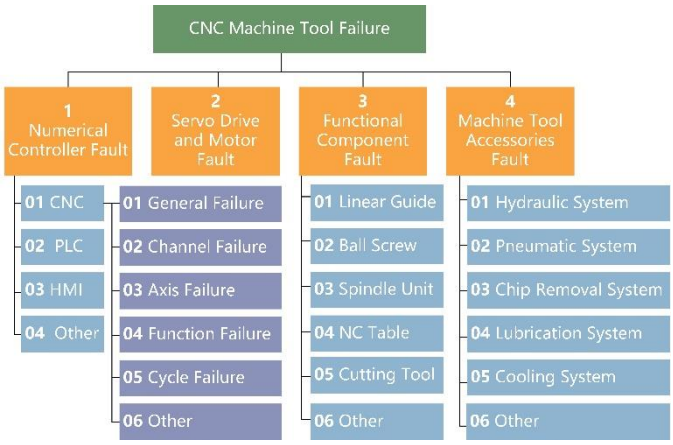

Fig. 8 The information model of the CNC machine tool fault

The information model of the CNC machine tool fault is shown in Fig. 8. Orange shows the fault classification of the CNC machine tool. CNC machine tool faults can be grouped into four categories: numerical controller fault, servo drive and motor fault, functional component fault, and machine tool accessories fault. The blue indicates the primary catalog of faults, and the purple marks the secondary catalog of faults. Coding the CNC machine tool fault depending on the information model, fault code a total of 9 bits, the first bit shows the classification of the fault, the 2-3 bits and 4-5 bits indicate the fault of the primary and the secondary catalog, respectively, which are coded from 01. The fault that cannot be categorized will be the shelter category "Other". Bits 6-9 indicate the specific fault information and are coded from 0001 onward. If there is no corresponding primary or secondary fault directory under the fault category, the corresponding position is coded with 00. To facilitate understanding, Fig. 9 gives an example of the code for "Excessive CNC Temperature".

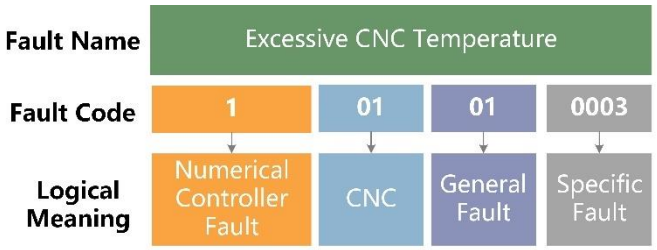

Fig. 9 Example of fault coding

5.2.2 Cyber layer: modeling and deployment of PHM

The CNC machine tool is a combination of different subsystems and components, and each subsystem and subcomponent has its own mechanism model for fault diagnosis and prediction. As an essential component of CNC machine tools, the health status of tools directly affects the quality and accuracy of machining, and the usage of the cutting tools is one of the most critical concerns of PHM service users. Here we take cutting tool wear diagnosis and prediction as an example and introduce the method of establishing and deploying the DT model of machine tool fault.

There are many possible disturbing factors for cutting tool wear, and each tool has a specific wear curve. Direct measurement of tool wear during machining is not feasible due to tool occlusion and cutting fluid. In addition, there is no accurate physically-based model for cutting tool wear assessment. Traditional tool wear diagnosis is usually based on workers' experience, and this manner may result untimely or excessive maintenance. To address the above problems, Sun et al. [48] proposed using a residual network (ResNet) to monitor tool wear in real-time. The unique shortcut structure of the ResNet effectively solves the degradation problem that occurs when the neural network model is too deep. On top of this, this paper introduces a temporal convolutional network (TCN) to predict tool wear, considering the importance of predicting tool conditions for data-driven smart manufacturing. As shown in Fig. 10, the tool's PHM service is supported by the combination of ResNet-TCN. The model is trained in the cloud using the offline dataset, and the subtasks are divided. Subtask 1 is the data preprocessing module consisting of *Down Sample* and *Wavelet Transform*. Subtasks 2-21 are the wear monitoring stage, containing one *Post-activation residual block*, eighteen identical *Pre-activation residual blocks*, and one *Dense block*. Similarly, subtasks 22-25 are the wear prediction stage, containing three identical *TCN* blocks and one *Dense* layer. Compared with a fully neural network layered task segmentation, this method segments subtasks according to the data processing stages and residual blocks. It avoids the situation where shortcuts cannot be split, and it can effectively simplify the segmentation process and provide the algorithm with fewer candidate segment points, thereby accelerating the offloading speed.

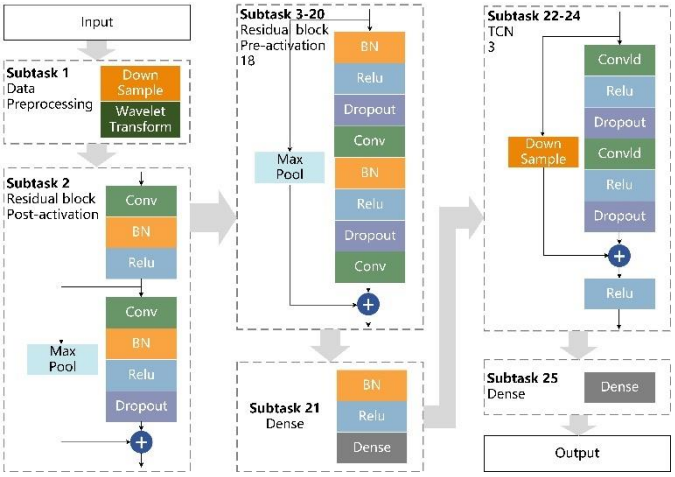

Fig. 10 The PHM model of cutting tool and subtask division

In this paper, [dataset] the IEEE PHM 2010 challenge dataset [49] was used to train the model of cutting tool wear. The dataset records real-time data from a Röders Tech RFM760 high-speed CNC milling machine machining a stainless-steel workpiece (HRC-52) under the same machining conditions, using approximately 315 cutting cycles for the six three-spline tungsten carbide ball-tip milling tools, and collecting X-, Y-, and Z-axis forces, three-axis accelerations, and acoustic emission sensor signals during machining, all with a data sampling rate of 50 kHz.

The trained model of the cutting tool's PHM is deployed according to the offloading strategy proposed in Section 4. First, the execution delay of each subtask on the nodes and servers is initialized. The execution delay of subtasks needs to be initialized only once for the same hardware because it is determined solely by the computational power of the hardware devices, and the delay of each subtask can be obtained from actual measurements. Fig. 11 shows each subtask's execution delay, data transmission delay, and output data dimension.

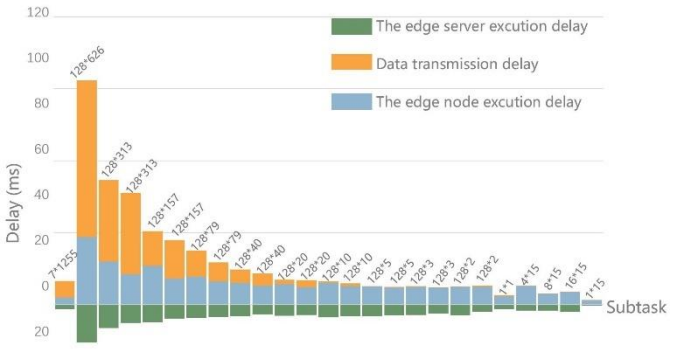

Fig. 11 Execution delay, transmission delay, and output data dimension of subtasks

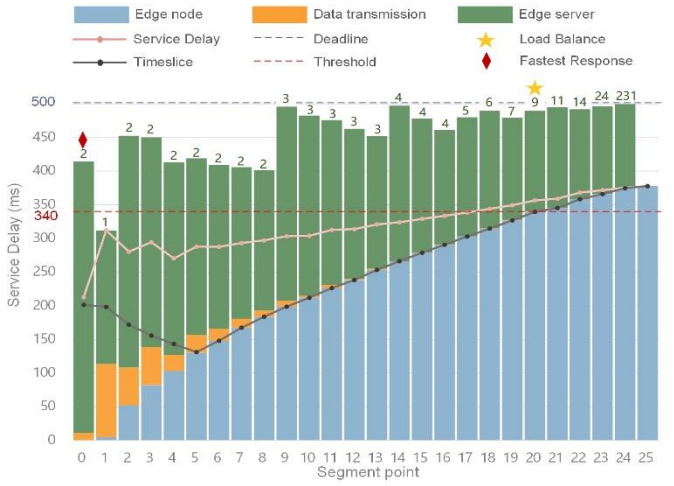

Fig. 12 The offloading segment points

The task as of the deadline is set to 500ms, and the throughput rate threshold $\theta = 30\%$. The output offloading segment points are shown in Fig. 12 according to the two offloading strategies, respectively. Blue indicates the node execution delay, orange indicates the task transmission delay, and green indicates the server execution delay. **Algorithm 1** uses a single server connected to a single node and aims at the fastest response for task offloading, so the shortest service delay is selected as the offloading segment point. In Fig. 12, the pink line points indicate the service delay, and red diamonds mark the offloading segment point output by **Algorithm 1**. In other words, when the relationship of server-to-node is 1-to-1, the fastest response can be obtained by offloading the whole task to the server for execution.

Algorithm 2 aims to balance the load. The candidate segment point with the maximum number of nodes accessible to the server is selected as the offloading segment point while satisfying the real-time demands of the task and throughput. The timeslice describes the throughput and is indicated by a dashed black dot with a throughput threshold corresponding to a 340ms time slice. The number above the bar indicates the maximum number of accessible nodes of the server corresponding to this candidate slice point. The yellow star symbol marks the offload segment point produced by **Algorithm 2**, i.e., after the first 20 subtasks are left to be processed by the node, the remaining subtasks are offloaded to the server for processing. At this point, the server can access 9 nodes, and the throughput rate is 30.58%.

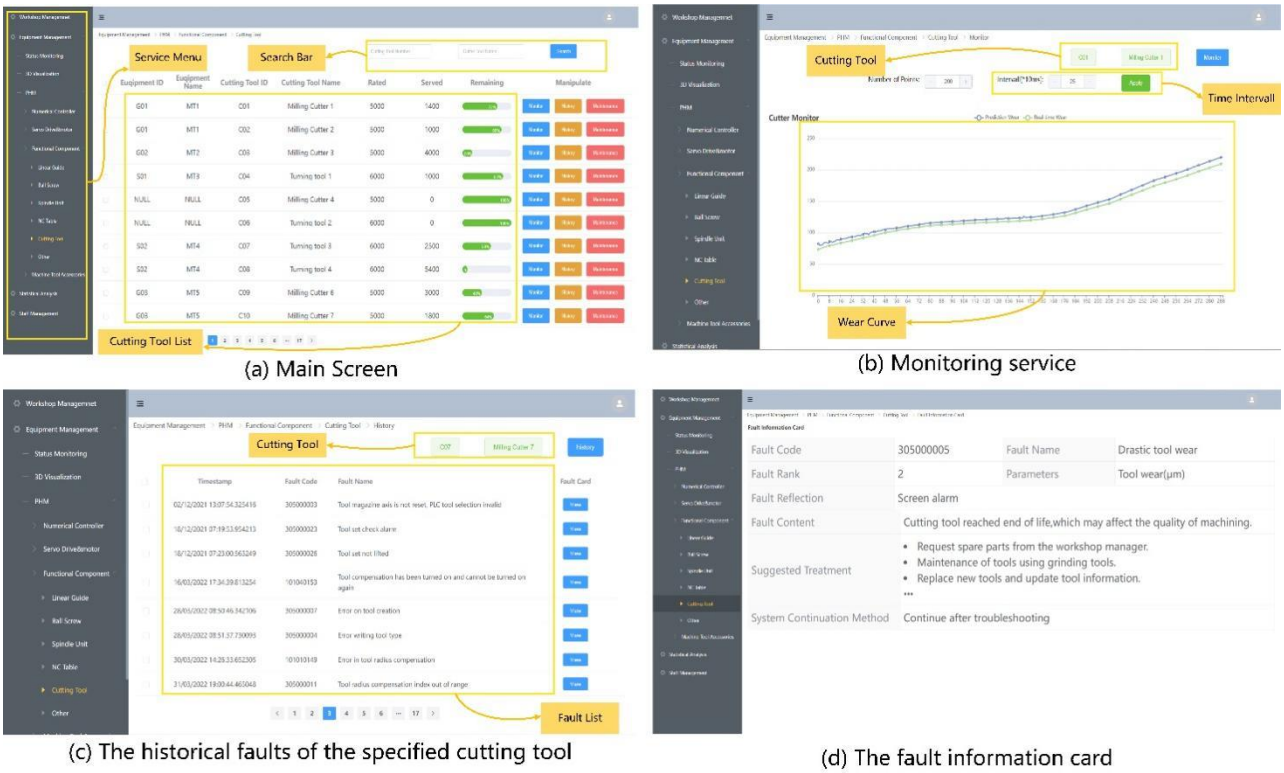

Fig. 13 UI of the application

5.3 Application layer: PHM of cutting tool

This section shows the web-based application where the client includes a web browser through which the user accesses the PHM application services for the cutting tools provided. Fig. 13 shows the user interface (UI) of the application we developed, with service modules such as workshop management, equipment management, statistical analysis, and staff management on the left. Furthermore, the PHM module for cutting tools is shown here as an example. The main screen (Fig.13 (a)) displays tool information, including the ID and name of the equipment it belongs to, the ID, name, rated lifespan, and served lifespan of the cutting tool. Meanwhile, the remaining lifespan of the cutting tool through a percentage stacked bar chart, which can help the user understand the status of the tool more intuitively. The users can search by ID or cutting tool name to display information about the specified cutting tool and use the corresponding "Manipulate" button to monitor, maintain, or view the cutting tool's history faults.

As shown in Fig. 13, the PHM of a tool provides three main functional modules, and Fig.13(b) shows the monitoring service. By searching for the tool ID or name in the search field, the user can view the graph of the cutting tool wear at the bottom of the page. The blue line in the graph shows the actual wear value, while the orange line shows the prediction wear value. Fig.13 (c) shows the user a list of historical faults of the specified cutting tool, including the fault timestamp, fault code, and fault name. By tapping the "View" button, the user can see the related fault information card. The fault information card is shown in Fig.13 (d) with "Drastic tool wear" as an example. The cutting tool wear threshold has been established as the event's trigger condition. When the wear value reaches the threshold, the CNC system provides the fault code "305000005" through the PLC, the screen alarm, and the client prompts the operator "Cutting tool reached the end of life, which may affect the quality of machining.", and gives suggested treatment to help the operator deal with the fault quickly to prevent further damage.

6 Conclusion and future work

The era of big data-driven manufacturing has urgently demanded intelligence, interconnectivity, adaptability, and autonomy of machine tools. As a typical representative of Machine Tool 4.0, CPMT realizes the feedback control of CNC machine tools by building MTDT, helps humans understand the state of machine tools, improves product quality, and to some extent, completes the transformation of machine tool digitization and service. Based on the existing CPMT conceptual model and the existing resources of industrial field devices, this paper proposes a CPMT architecture based on cloud-edge collaboration. It transfers the establishment of the DT model and data processing from the cloud to the edge and provides microservice interfaces to the edge, thus bringing better service quality to users in terms of low power consumption, privacy protection, and fast response. Then, task offloading methods for speed of response and task offloading methods for load balancing are proposed considering the limitations of on-premises computing resources and the needs of latency-sensitive applications. Finally, this paper presents an application case of cloud-edge collaboration-based CPMT to provide PHM services for machine tools. The work deploys hardware devices according to the service requirements, builds the corresponding MTDT model, verifies the feasibility of the proposed cloud-edge collaboration-based CMPT, and demonstrates its benefits and potential through the PHM service for cutting tools.

The objective of this work is to build CPMT collaboratively on the cloud side. The current research work has made a preliminary discussion on the construction and deployment of the CPMT model, and the proposed task offloading method needs further enrichment and improvement. The directions for future work are summarized as follows.

1) Consider distributed task offloading strategies for large-scale edge networks to ensure algorithm scalability and prevent single-point failures from affecting the entire system.

2) Accommodate the variability in resource allocation of edge servers, i.e., nodes can only offload tasks to the edge server containing the required computing resources.

3) Provide richer application services, integrate more artificial intelligence algorithms into the MTDT, and develop mobile applications to accommodate different types of HMIs.

4) Explore the operating mechanism of CPMT in physical information production systems (CPPS) and consider horizontal integration for autonomous cooperation between devices.

Acknowledgments

This work was supported by the High-level Innovation

and Entrepreneurship Team in Liaoning Province under Grant XLYC1902091.

References

[1] T. Kjellberg, A. von Euler-Chelpin, M. Hedlind, M. Lundgren, G. Sivard, D. Chen, The machine tool model-A core part of the digital factory, CIRP Ann. - Manuf. Technol. 58 (2009) 425–428. https://doi.org/10.1016/j.cirp.2009.03.035.

[2] C. Liu, H. Vengayil, R.Y. Zhong, X. Xu, A systematic development method for cyber-physical machine tools, J. Manuf. Syst. 48 (2018) 13–24. https://doi.org/10.1016/j.jmsy.2018.02.001.

[3] X. Xu, Machine Tool 4.0 for the new era of manufacturing, Int. J. Adv. Manuf. Technol. 92 (2017) 1893–1900. https://doi.org/10.1007/s00170-017-0300-7.

[4] C. Liu, X. Xu, Cyber-physical Machine Tool - The Era of Machine Tool 4.0, Procedia CIRP. 63 (2017) 70–75. https://doi.org/10.1016/j.procir.2017.03.078.

[5] C.K.M. Lee, B. Lin, K.K.H. Ng, Y. Lv, W.C. Tai, Smart robotic mobile fulfillment system with dynamic conflict-free strategies considering cyber-physical integration, Adv. Eng. Informatics. 42 (2019). https://doi.org/10.1016/j.aei.2019.100998.

[6] F. Tao, J. Cheng, Q. Qi, IIHub: An industrial internet-of-things hub toward smart manufacturing based on cyber-physical system, IEEE Trans. Ind. Informatics. 14 (2018) 2271–2280. https://doi.org/10.1109/TII.2017.2759178.

[7] B. Liu, Y. Zhang, G. Zhang, P. Zheng, Edge-cloud orchestration driven industrial smart product-service systems solution design based on CPS and IIoT, Adv. Eng. Informatics. 42 (2019) 100984. https://doi.org/10.1016/j.aei.2019.100984.

[8] M. Zolanvari, M.A. Teixeira, L. Gupta, K.M. Khan, R. Jain, Machine Learning-Based Network Vulnerability Analysis of Industrial Internet of Things, IEEE Internet Things J. 6 (2019) 6822–6834. https://doi.org/10.1109/JIOT.2019.2912022.

[9] M. Zhang, F. Tao, B. Huang, A. Liu, N. Anwer, A.Y.C. Nee, Digital twin data : methods and key technologies [version 1 ; peer review : awaiting peer review], (2021).

[10] I.E. Computing, Industrial Edge Computing : Vision and Challenges, Inf. Control. 50 (2021) 257–274.

[11] C. Ding, A. Zhou, Y. Liu, R. Chang, C.H. Hsu, S. Wang, A Cloud-Edge Collaboration Framework for Cognitive Service, IEEE Trans. Cloud Comput. 7161 (2020). https://doi.org/10.1109/TCC.2020.2997008.

[12] Edge computing Consortium and Alliance of Industrial Internet: White Paper on Edge Computing and Cloud Computing Collaboration[EB/OL], Http://Www.Ecconsortium.Org/Uploads/File/20190221/1550718911180625. (n.d.).

[13] M. Huang, W. Liu, T. Wang, A. Liu, S. Zhang, A Cloud-MEC Collaborative Task Offloading Scheme with Service Orchestration, IEEE Internet Things J. 7 (2020) 5792–5805. https://doi.org/10.1109/JIOT.2019.2952767.

[14] C. Liu, P. Zheng, X. Xu, Digitalisation and servitisation of machine tools in the era of Industry 4.0: a review, Int. J. Prod. Res. (2021). https://doi.org/10.1080/00207543.2021.1969462.

[15] C. Deng, R. Guo, P. Zheng, C. Liu, X. Xu, R.Y. Zhong, From Open CNC Systems to Cyber-Physical Machine Tools: A Case Study, Procedia CIRP. 72 (2018) 1270–1276. https://doi.org/10.1016/j.procir.2018.03.110.

[16] C. Liu, X. Xu, Q. Peng, Z. Zhou, MTConnect-based Cyber-Physical Machine Tool: A case study, Procedia CIRP. 72 (2018) 492–497. https://doi.org/10.1016/j.procir.2018.03.059.

[17] Z. Zhu, X. Xu, User-centered information provision of cyber-physical machine tools, Procedia CIRP. 93 (2020) 1546–1551. https://doi.org/10.1016/j.procir.2020.04.091.

[18] Z. Cao, P. Zhou, R. Li, S. Huang, D. Wu, Multiagent Deep Reinforcement Learning for Joint Multichannel Access and Task Offloading of Mobile-Edge Computing in Industry 4.0, IEEE Internet Things J. 7 (2020) 6201–6213. https://doi.org/10.1109/JIOT.2020.2968951.

[19] J. Liu, Y. Mao, J. Zhang, K.B. Letaief, Delay-optimal computation task scheduling for mobile-edge computing systems, IEEE Int. Symp. Inf. Theory - Proc. 2016-August (2016) 1451–1455. https://doi.org/10.1109/ISIT.2016.7541539.

[20] V.J.& K. Patil, A Survey on Energy-Efficient Task Offloading and Virtual Machine Migration for Mobile Edge Computation, https://doi.org/10.1007/978-981-16-2937-2_22.

[21] H. Tang, C. Li, J. Bai, J.H. Tang, Y. Luo, Dynamic resource allocation strategy for latency-critical and

computation-intensive applications in cloud–edge environment, Comput. Commun. 134 (2019) 70–82. https://doi.org/10.1016/j.comcom.2018.11.011.

[22] Q. Qi, D. Zhao, T.W. Liao, F. Tao, Modeling of cyber-physical systems and digital twin based on edge computing, fog computing and cloud computing towards smart manufacturing, ASME 2018 13th Int. Manuf. Sci. Eng. Conf. MSEC 2018. 1 (2018). https://doi.org/10.1115/MSEC2018-6435.

[23] C. Yang, S. Lan, L. Wang, W. Shen, G.G.Q. Huang, Big data driven edge-cloud collaboration architecture for cloud manufacturing: A software defined perspective, IEEE Access. 8 (2020) 45938–45950. https://doi.org/10.1109/ACCESS.2020.2977846.

[24] J. Zhang, C. Deng, P. Zheng, X. Xu, Z. Ma, Development of an edge computing-based cyber-physical machine tool, Robot. Comput. Integr. Manuf. 67 (2021) 102042. https://doi.org/10.1016/j.rcim.2020.102042.

[25] P. Lou, S. Liu, J. Hu, R. Li, Z. Xiao, J. Yan, Intelligent Machine Tool Based on Edge-Cloud Collaboration, IEEE Access. 8 (2020) 139953–139965. https://doi.org/10.1109/ACCESS.2020.3012829.

[26] F.T. Chenyuan Zhang, Evaluation index system for digital twin model, Comput. Integr. Manuf. Syst. 27(8) (2021) 2171–2186. https://doi.org/10.13196/j.cims.2021.08.001.

[27] M. Schluse, M. Priggemeyer, L. Atorf, J. Rossmann, Experimentable Digital Twins-Streamlining Simulation-Based Systems Engineering for Industry 4.0, IEEE Trans. Ind. Informatics. 14 (2018) 1722–1731. https://doi.org/10.1109/TII.2018.2804917.

[28] D. Twin, Z. Lv, S. Xie, Artificial intelligence in the digital twins : State of the art , challenges , and future research topics [version 1 ; peer review : awaiting peer review] Zhihan Lv, (2021) 1–20.

[29] C. Liu, P. Jiang, W. Jiang, Web-based digital twin modeling and remote control of cyber-physical production systems, Robot. Comput. Integr. Manuf. 64 (2020) 101956. https://doi.org/10.1016/j.rcim.2020.101956.

[30] M. Bandara, F.A. Rabhi, Semantic modeling for engineering data analytics solutions, Semant. Web. 11 (2020) 525–547. https://doi.org/10.3233/SW-190352.

[31] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, J. Zhang, Edge Intelligence: Paving the Last Mile of Artificial Intelligence With Edge Computing, Proc. IEEE. (2019) 1–25. https://doi.org/10.1109/JPROC.2019.2918951.

[32] M. Kumar, S.C. Sharma, A. Goel, S.P. Singh, A comprehensive survey for scheduling techniques in cloud computing, J. Netw. Comput. Appl. 143 (2019) 1–33. https://doi.org/10.1016/j.jnca.2019.06.006.

[33] Y. Lu, X. Xu, Cloud-based manufacturing equipment and big data analytics to enable on-demand manufacturing services, Robot. Comput. Integr. Manuf. 57 (2019) 92–102. https://doi.org/10.1016/j.rcim.2018.11.006.

[34] W.L. Weisong Shi, Hui Sun, Jie Cao, Quan Zhang, Edge Computing--An Emerging Computing MOdel for the Internet of Everything Era, J. Comput. Res. Dev. 54(05) (2017) 907–924. https://doi.org/10.7544/issn1000-1239.2017.20160941.

[35] M. Satyanarayanan, The emergence of edge computing, Computer (Long. Beach. Calif). 50 (2017) 30–39. https://doi.org/10.1109/MC.2017.9.

[36] X.Z. Yang Liu, Research on industrial digital twin technology system and key technologies, Inf. Commun. Technol. Policy. 47 (2021) 8–13. https://doi.org/10.12267/j.issn.2096-5931.2021.01.003.

[37] H. Yu, D. Yu, Y. Hu, C. Wang, Research on CNC Machine Tool Monitoring System Based on OPC UA, Proc. 31st Chinese Control Decis. Conf. CCDC 2019. (2019) 3489–3493. https://doi.org/10.1109/CCDC.2019.8832877.

[38] A. Gaikwad, R. Yavari, M. Montazeri, K. Cole, L. Bian, P. Rao, Toward the digital twin of additive manufacturing: Integrating thermal simulations, sensing, and analytics to detect process faults, IISE Trans. 52 (2020) 1204–1217. https://doi.org/10.1080/24725854.2019.1701753.

[39] et al. Tao Fei, Liu Weiran, Hu Tianliang, Five-dimension digital twin model and its ten applications, Comput. Integr. Manuf. Syst. 25 (2019) 1–18.

[40] X. Wang, Y. Wang, F. Tao, A. Liu, New Paradigm of Data-Driven Smart Customisation through Digital Twin, J. Manuf. Syst. 58 (2021) 270–280. https://doi.org/10.1016/j.jmsy.2020.07.023.

[41] Y. Lu, C. Liu, K.I.K. Wang, H. Huang, X. Xu, Digital

Twin-driven smart manufacturing: Connotation, reference model, applications and research issues, Robot. Comput. Integr. Manuf. 61 (2020) 101837. https://doi.org/10.1016/j.rcim.2019.101837.

[42] E. al. Fei Tao, He Zhang, Qinglin Qi, Theory of digital twin modeling and its application, Comput. Integr. Manuf. Syst. 27 (2021) 1–15. https://doi.org/10.13196/j.cims.2021.01.001.

[43] GB/T 39561.4-2020, Interconnection and interoperation of numerical control equipment— Part 4: Object dictionary of numerical control machine tools [S]., (n.d.)., (n.d.).

[44] L. Wang, C. Wu, W. Fan, A Survey of Edge Computing Resource Allocation and Task Scheduling Optimization, Xitong Fangzhen Xuebao / J. Syst. Simul. 33 (2021) 509–520. https://doi.org/10.16182/j.issn1004731x.joss.20-0584.

[45] Y. Kang, J. Hauswald, C. Gao, A. Rovinski, T. Mudge, J. Mars, L. Tang, Neurosurgeon: Collaborative intelligence between the cloud and mobile edge, ACM SIGPLAN Not. 52 (2017) 615–629. https://doi.org/10.1145/3037697.3037698.

[46] A. Halaas, B. Svingen, M. Nedland, P. Sætrom, O. Snøve, O.R. Birkeland, A recursive MISD architecture for pattern matching, IEEE Trans. Very Large Scale Integr. Syst. 12 (2004) 727–734. https://doi.org/10.1109/TVLSI.2004.830918.

[47] C. Liu, H. Zhu, D. Tang, Q. Nie, T. Zhou, L. Wang, Y. Song, Probing an intelligent predictive maintenance approach with deep learning and augmented reality for machine tools in IoT-enabled manufacturing, Robot. Comput. Integr. Manuf. 77 (2022) 102357. https://doi.org/10.1016/j.rcim.2022.102357.

[48] H. Sun, J. Zhang, R. Mo, X. Zhang, In-process tool condition forecasting based on a deep learning method, Robot. Comput. Integr. Manuf. 64 (2020) 101924. https://doi.org/10.1016/j.rcim.2019.101924.

[49] PHM Society, PHM Society Conference Data Challenge, 2010 https://www.phmsociety.org/competition/phm/10 2010 (accessed 20 December 2018)., (n.d.).