1415 N CHERRY AVE
CHICAGO, IL 60642
(312) 281-6900
DMDII.ORG
DMDII@UILABS.ORG

DMDII
a UI LABS Collaboration
DIGITIZING AMERICAN MANUFACTURING

DMDII FINAL PROJECT REPORT

DMDII 15-14-01:
Cloud-Enabled Machines with Data-Driven Intelligence

Principle Investigator: Janis Terpenny/jpt5311@psu.edu

Project Team Lead: Penn State University

Project Designation: DMDII-15-14-01

UI LABS Contract Number

Project Participants: Penn State University, Case Western Reserve University, University of Central Florida, General Electric Company

DMDII Funding Value: $749,625

Project Team Cost Share: $749,939

Award Date: 11-12-2016

Completion Date: 09-30-2018

SPONSORSHIP DISCLAIMER STATEMENT: This project was completed under the Cooperative Agreement W31P4Q-14-2-0001, between U.S. Army - Army Contracting Command - Redstone and UI LABS on behalf of the Digital Manufacturing and Design Innovation Institute. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Department of the Army.

TABLE OF CONTENTS

1. Executive Summary

Over the past few decades, manufacturers have been faced with an increasing need for the development of low cost and scalable intelligent manufacturing machines that are capable of diagnosing the root cause of identified defects, predicting their progression, and forecasting maintenance actions proactively to minimize unexpected machine down times. Current limitations of manufacturing machines are as follows:

- Lack of scalable and reliable data acquisition systems that are capable of capturing the health condition of machines in real-time;
- Lack of effective and efficient algorithms and computing capacity that allow fault diagnosis, predictive maintenance, and prognosis.

The advances in cloud computing, Internet of Things (IoT), cyber-physical systems (CPS), and artificial intelligence automatic have the potential to enable fault and failure detection, self-diagnosis, and predictive maintenance. The overcharging goal of this research is to integrate cloud computing, low-cost sensors, machine learning, and signal processing techniques into manufacturing equipment for online machine and process monitoring, diagnosis, and prognosis. The specific objectives of this project are as follows:

- Develop a generic framework for cloud-based online machine and process monitoring, diagnosis, and prognosis;
- Develop a private cloud-based data acquisition system that collects massive data from machines and processes using the ICT infrastructure that is solely operated within a corporate firewall;
- Develop a hybrid cloud platform that integrates the cloud-based data acquisition system with a public high-performance cloud computing system;
- Develop parallel and distributed machine learning algorithms for online diagnosis and prognosis in additive and subtractive manufacturing as well as motors and bearings.

Specifically, an interoperable sensing system consisting of "drop-in" sensor nodes, a gateway device, and pre-configured "protocol adapters" for plug-and-play fieldbus communications have been developed to address machine connectivity and data collection. A container-based private cloud infrastructure that provides a petabyte-scale, high performance, and low latency distributed file system as well as a scalable cloud computing environment with real-time stream analytics, data visualization, and parallel machine learning tools have been developed for processing high volume and high-speed data streams. A sparse representation-based classification method has been developed and implemented in the hybrid cloud system to diagnose multiple fault sources. A particle filter-based approach has been developed to predict the system performance and remaining useful life of manufacturing machines. The final project deliverables include:

- An interoperable data acquisition and on-premise cloud computing platform providing scalable data collection and processing for hundreds of manufacturing machines on factory floors;
- A public cloud platform integrated with on-premise private cloud for processing real-time data streams, executing parallel machine learning algorithms, generating big data analytics, and visualizing data;
- A set of experimentally tested algorithms enabling data-driven intelligence for online machine fault diagnosis and prognosis in various types of manufacturing machines and processes, executable on a hybrid cloud computing platform.

2. Project Review

2.1 Background

Manufacturers have been faced with the increasing need for hardware and software tools that efficiently collect and process large volumes of data generated from machines and manufacturing processes as well as

algorithms that effectively diagnose the root cause of identified defects, predict their progression, and forecast maintenance action proactively to minimize unexpected machine down times. From a hardware perspective, many manufacturing machines and processes are still insufficiently monitored due to the lack of sensors that meet application requirements in terms of space, packaging, cost, functionality, environmental effects, etc., for plug and play installation and in-situ data acquisition. From a software perspective, databases containing data collected over a large time span from diverse machines and processes, ICT infrastructures with sufficient computational capacity and bandwidth for processing high-speed and high-volume data streams, as well as algorithms for parallel and distributed big data processing that are implementable on the factory floors, are urgently needed.

In light of cloud computing, a new manufacturing paradigm, namely cloud-based manufacturing (CBM), has been introduced. CBM refers to a service-oriented digital manufacturing model that enables the acquisition and analysis of machine- and process-related data by leveraging cloud computing, IoT, and CPS. CBM has the following unique advantages:

- Ubiquitous and instant remote access to near real-time data without spatial constraints.
- Secure and high volume data storage. Cloud computing provides manufacturers with reliable, secure, scalable, and economical storage of massive static and dynamic data. The advantage of cloud storage is that it delivers high performance, low-latency communication for I/O intensive workloads such as high-speed data collection and processing.
- Scalable, high performance computing (HPC). Compared to the traditional manufacturing paradigms, CBM can significantly increase computing capacity by providing multiple- and many-core processors to complement high-volume storage and high-speed I/O interconnects. This allows manufacturers to scale up computing capacity rapidly and cost effectively when computing needs increase and then scale down as demands decrease.
- Big data analytics. Enabled by parallel and distributed computing, data mining and machine learning algorithms can be developed that enable manufacturers to process and manage massive data streams on a cloud-based computing platform. Specifically, CBM employs an open-source software framework that supports data-intensive distributed applications.

2.2 Problem Statement and DMDII Relevance

Problem Statement: Over the past few decades, one of the primary problems faced by both small- and medium-sized manufacturers (SMMs) and large original equipment manufacturers (OEMs) is how to develop new machines with intelligence as well as retrofit legacy machines with intelligence so that in-process, remote monitoring, diagnosis, prognosis, and self-correction can be automatically performed. Traditional monitoring systems have limitations in accessing and synchronizing massive data sets acquired from multiple machines and processes in a distributed environment as well as processing large volumes and high-speed data streams. With the advancement of parallel computing and intelligent sensing systems, cloud computing, IoT, and CPS have been increasingly recognized as promising technological solutions to the problem. Although both academia and industry are motivated to explore these advanced technologies for manufacturing, little work has been reported on integrating cloud computing, smart sensors, and parallel data mining and machine learning into online machine and process monitoring, diagnosis, and prognosis. Addressing this gap, the proposed research aims to answer the following questions:

- What structure would be required to implement a generic framework for cloud-based online machine and process monitoring, diagnosis, and prognosis?
- How can massive data in a distributed environment be collected and analyzed by potentially unlimited, scalable, cost-effective, high performance computing platforms that provide ubiquitously accessible storage and are reliable, while maintaining the ability to control data and mitigate risks to the infrastructure?
- How can data mining and machine learning algorithms be parallelized so that computationally intensive methods in diagnosis and prognosis be performed efficiently and remotely?

To answer these questions, the specific objectives of this project are as follows:

- Develop a generic framework for cloud-based online machine and process monitoring, diagnosis, and prognosis;
- Develop a private cloud-based data acquisition system that collects massive data from machines and processes using the ICT infrastructure that is solely operated within a corporate firewall;
- Develop a hybrid cloud platform that integrates the cloud-based data acquisition system with a public high-performance cloud computing system;
- Develop parallel and distributed machine learning algorithms for online diagnosis and prognosis in additive and subtractive manufacturing as well as motors and bearings.

DMDII Relevance: As stated in the project call, the goal of 15-14 is "to implement machine intelligence into manufacturing machines as well as promote the adoption of relevant standards for sensing systems, sensing system communications and integration into manufacturing machines and systems." The scope of this project call includes "both new machines having built-in sensors and intelligence as well as legacy machines and systems that have been retrofitted with sensors and intelligence." Based on the aforementioned objectives and scope of work, the proposed project is highly relevant to the DMDII-15-14 project call from the following perspectives:

- The first research objective is aligned with the increasing demand for developing a generic framework for online machine and process monitoring, diagnosis, and prognosis. Specifically, because of capabilities enabled by cloud computing, the generic framework of our work could help manufacturers develop low cost and scalable intelligent systems with plug-and-play interoperability;
- The second research objective is aligned with the integration of sensors and sensor networks into legacy machines and general purpose CNC machining centers of a manufacturer so that massive online data generated from machines and processes can be collected by the private cloud-based data acquisition system.
- The third and fourth research objectives are aligned with the goal to introduce machine intelligence into legacy machines and general purpose CNC machining centers. By integrating a private cloud-based data acquisition hardware system, with the public HPC cloud infrastructure, computational-intensive tasks such as training massive datasets and data analytics can be performed on scalable, secure, and high performance cloud computing platforms to transform legacy and conventional stand-alone machines on the factory floor into cloud-based machines with data-driven intelligence.

2.3 Methodology

Fig. 1 illustrates an architecture of a cloud-enabled machine and process monitoring, diagnostics, and prognostics system. An interoperable gateway device collects real-time data streams from factory floors through sensor networks, protocol and sensor adapters, and I/O connectors. A private cloud platform stores, screens, and cleans the data streams. A public HPC cloud performs computationally-intensive operations, including executing machine learning algorithms, generating big data analytics, and visualizing data analytics, on the pre-processed data streams. Once diagnostic and prognostic models are created using the pre-processed training data, these models are executed in an on-premise private cloud platform for online diagnosis and prognosis.

The public cloud performs the following tasks:

- Predictive model training using machine learning algorithms
- Cloud streaming analytics using PHM algorithms
- Cloud predictive model inferencing

The private cloud performs the following tasks:

- Machine data collection and preprocessing

- On-premise high-volume streaming analytics using PHM algorithms
- On-premise predictive model inferencing

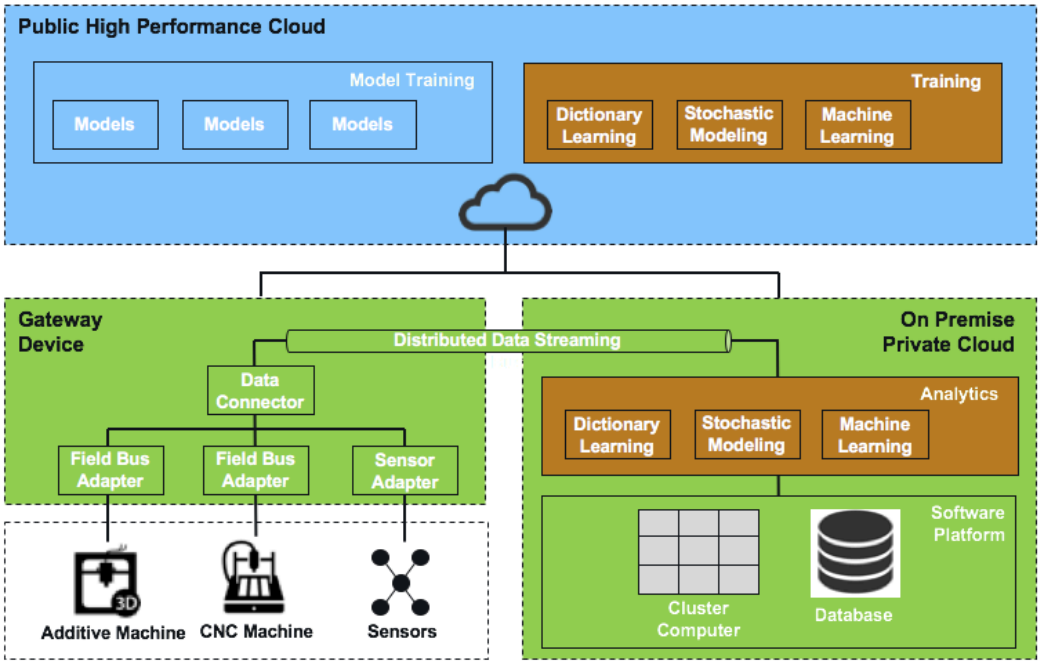


Figure 1. An architecture of cloud-enabled machine monitoring, diagnostics and prognostics system

Fig. 2 illustrates a computational framework for data-driven predictive modeling. The framework consists of data collection, data processing, and modeling training and validation.

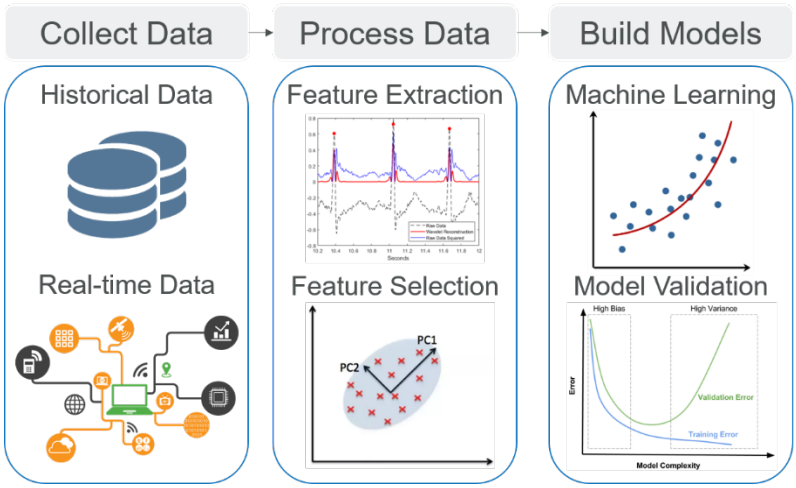


Figure 2. A computational framework

2.4 Research Tasks

Fig. 3 shows an overview of the proposed research tasks.

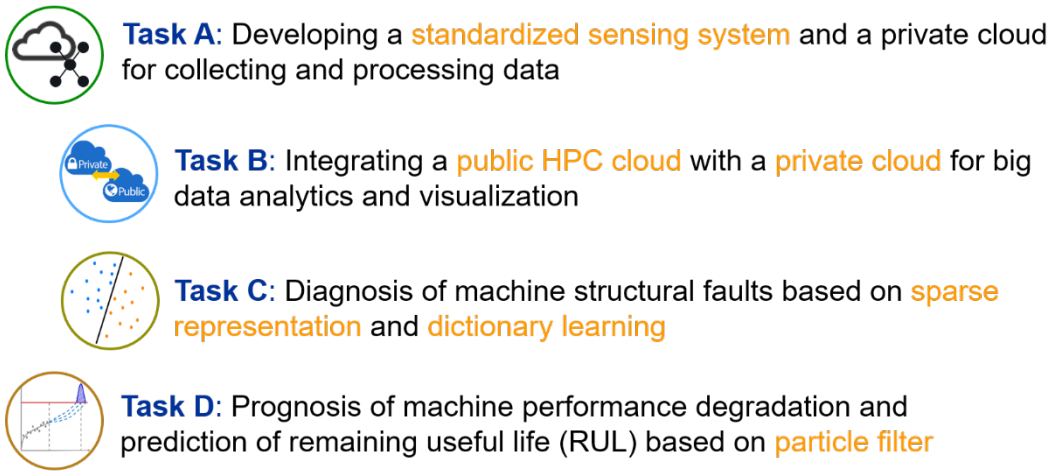


Figure 3. An overview of research tasks

2.4.1 Task A: Developing a standardized sensing system and a private cloud for collecting and processing data

Current problem: Machine and process data are essential to successful on-line monitoring, diagnosis and prognosis. In older factories, legacy machines either do not have enough sensors or do not have a way to expose sensor data to external applications. On the other hand, machines in a modern factory are equipped with rich sensors and are connected to other machines through standard fieldbuses. However, the problem lies in the diversity of communication protocols that range from simple RS485, Modbus, to modern OPC-UA, PROFINET, and to newly adopted MTConnect. Custom built hardware and software are often required to interface with each of the machines. Once data are collected and aggregated, there is a lack of computing resource on factory floors to process data and make intelligent decisions. Computers associated with manufacturing machines are purposely designed for control tasks. Generic on-premise computing platforms can be added to the factory floors, but they are not architected to process large volumes of data in real-time. It is also expensive and difficult to scale them up to the level required for training highly iterative data-driven machine learning models and algorithms. Cloud computing platforms are perfectly suited for such tasks due to their scalability and elasticity. However, this requires sending large volumes of data to the cloud. For example, a typical medium-size GE factory with machines fully instrumented with sensors can generate up to 152,000 samples of data per second, or 13 billion samples per day. Even if the network speed and bandwidth can withstand this workload, the cost associated with it is often prohibitive.

Proposed Solution: To address the machine connectivity and data collection problem, an on-premises framework has been developed. This framework includes (1) an interoperable data acquisition system (DAS) that supports real-time, scalable, and plug-and-play data collection for both legacy and general-purpose machines and (2) a lightweight on-premises computing platform that can deploy Linux container-based software for running data-driven diagnostic and prognostic algorithms and visualization. The DAS consists of "drop-in" sensor nodes, data aggregation gateway devices, and plug-and-play "protocol adapters" for fieldbus communications. A few example sensor nodes and protocol adapters have been developed to demonstrate how this framework can be used. Raw sensor data collected from machines is streamed to the on-premises cloud platform, where they are pre-processed, analyzed, and visualized. The pre-processed data can also be used to train and evaluate models on a public HPC cloud. Once data-driven diagnostic and prognostic models are created, these models were executed on the private cloud for online machine and process diagnosis and prognosis. In this manner, the amount of data transmitted cross the private, public clouds and factory floors has been reduced. Linux container technology has been used as the core framework for this system. Linux containers provide a way to virtualization on both embedded and server devices. It is a practical solution to enable developers to develop and deploy new algorithms and

tools for data acquisition, remote monitoring, diagnosis, prognosis and visualization on many different types of devices.

2.4.2 Task B: Integrating a public HPC cloud with a private cloud for big data analytics and visualization

Current problem: Over the past few decades, in-house supercomputers have been playing an important role in a wide range of computational and data intensive fields such as computational fluid dynamics (CFD) and finite element analysis (FEA). However, very few organizations have access to in-house supercomputers due to extremely high initial and maintenance costs. Since cloud computing has come into existence in the late 2000s, there is an increasing need to develop low-cost cloud computing platforms that enable manufacturers to accelerate compute- and data-intensive workloads. Specifically, HPC clouds enable manufacturers to have on-demand, ubiquitous, and instant access to large volumes of data, advanced computing infrastructures, and application software with no upfront costs as well as process high-speed data streams and generate data analytics. While cloud computing has been applied into computer-aided design, CFD, and FEA, little work has been reported on applying cloud computing for online machine and process monitoring and real-time analytics for manufacturing. Currently, cloud computing makes extensive use of hypervisor-based virtual machines (VMs) that enable the software implementation of a physical computer that executes programs like a physical machine. However, the hypervisor-based virtualization technology virtualizes not only an application and the necessary binaries and libraries but also an entire guest operating system. The disadvantage of hypervisor-based virtualization is that system performance may degrade due to additional storage, memory, and I/O overhead incurred by virtualizing the entire operating system. Therefore, traditional hypervisor-based virtual machines have limitations on the ability to process high-speed data streams generated by manufacturing machines and processes. In addition, another limitation of current cloud computing results from its deployment models or cloud architectures. In general, the most commonly implemented cloud architectures include private and public clouds. The primary drawback of private clouds is that additional computing resources need to be added periodically to scale up existing computing capacity. Although public clouds have potentially unlimited computing resources, users have limited control over data because cloud providers own and operate cloud infrastructures at data centers.

Proposed Solution: To address the aforementioned issues, container technology and hybrid cloud architecture are proposed. Container-based virtualization is an approach to virtualization in which the virtualization layer of cloud computing systems runs as an application within the operating system. In container-based virtualization, the kernel of the operating system runs on the hardware node with several isolated guest virtual machines installed atop. The isolated guests are called containers. Container-based virtualization has the potential to provide a lightweight virtualization layer, which promises a near-native system performance. Therefore, container-based virtualization not only simplifies the access and deployment of application software, but also reduces overhead and provides better performance. In Task B, Microsoft Azure cloud platform provides a petabyte-scale, high performance, and low latency distributed file system that support I/O intensive workloads. Docker containers for Linux operating systems on Azure offer an enterprise-level container-based cloud for processing large volume and high- speed data streams. In addition, Azure real-time stream analytics along with Machine Learning and Power BI services enables cloud-based big data analytics and data visualization.

Moreover, a hybrid cloud that integrates a private cloud with a public HPC cloud (i.e., Microsoft Azure Cloud Computing Platform), is developed for online diagnosis and prognosis. The key benefits of the hybrid cloud are that it employs the existing on-premise private cloud and combines it with a public cloud so that the hybrid cloud enables manufacturers to gain control over their proprietary data and mitigate security risks while acquiring access to scalable public clouds for compute- intensive workloads. In the hybrid cloud, manufacturers store sensitive data on the private cloud platform while utilizing intelligence and analytics applications provided by Microsoft Azure. Azure provides a graphical tool and a large set of supervised

and unsupervised machining learning algorithms for managing machine learning processes and performing machine learning.

2.4.3 Task C: Diagnosis of machine structural faults based on sparse representation and dictionary learning

Taking advantage of cloud-enabled distributed computing capability for on-line diagnosis of manufacturing machines and processes, this project investigates spindles in CNC machines at GE as an example to demonstrate cloud-based capability for intelligent machine systems.

Current Problem: Techniques for spindle diagnosis can be categorized as physics-based and data- driven. In physics-based methods, an analytical model is assumed to describe spindle performance under thermal affect, preload, centrifugal force and gyroscopic moment. The model is established based on first principles, and provides a mathematical representation of system degradation mechanism due to fault initiation and propagation. Given the close link to spindle physics and deterministic nature, such models tend to be application specific and limited in modeling stochastic phenomena associated with spindle operations. Data-driven methods, in comparison, extract fault patterns from the acquired sensor data through statistical analysis and machine learning. As a result, they are directly reflective of the temporal progression of spindle dynamics that involves the inception and deterioration of defects. Prior research in data-driven methods for spindle diagnosis has focused on understanding spindle state through analysis of sensor data, e.g., vibration, acoustic emission, force and temperature. Specific algorithms investigated for spindle diagnosis include wavelet transform, empirical mode decomposition (EMD), artificial neural network (ANN), support vector machine (SVM), adaptive network based fuzzy inference system (ANFIS), etc. While successful for the reported studies, each technique has its limitations:

- Limited adaptability: in wavelet-based diagnosis methods, single basis has been commonly used. This limits the effectiveness of these methods in extracting complex fault components embedded in spindle signals. Furthermore, they are sensitive to parameters chosen when performing computations. The lack of general guidelines limits the process of optimal parameters selection.
- Limited representation of nonlinear signals: in EMD-based methods, multiple frequency components can be included in the intrinsic mode functions. As a result, components in the signals reflecting system nonlinearity may not be completely extracted and properly separated. This limits the effectiveness of EMD in multi-frequency information representation for spindle diagnosis.
- Dependence on signal features: intelligent classification methods such as ANN, SVM and ANFIS are based on signal features. A common drawback is that features may not be reflective of the physical information contained in the signals. As a result, the effectiveness of the extracted features in revealing the actual state of the spindle may be limited for accurate diagnosis.

Developed Solution: To address the above limitations in spindle diagnosis, a diagnosis method based on dictionary learning and sparse classifier has been developed in this project (see Fig. 4). The advantage of the proposed model is that raw signal from sensors can be expressed sparsely, contributing to data dimensional reduction, enhancing efficiency in transmission for massive data in cloud-based diagnosis framework and facilitating fault-related pattern recognition. Two steps are involved in the developed method: off-line training and on-line diagnosis. For off-line training, historical sensing data obtained from q fault categories are first collected and processed using dictionary learning for fault characterization and fault-related pattern recognition. Parallel computing is carried out to process the signals from different fault categories simultaneously with each being assigned to a processor. Subsequently, the learned dictionaries, which contain fault-related information, is used to construct multi-fault sparse classifiers. The on-line diagnosis is performed in MapReduce-based parallel framework, data form each sensor are input to the corresponding multi-fault classifier to identify spindle fault. Similarly, the computation of each classifier is assigned to individual processor. When the results from each classifier are obtained, they're fused to make

the final diagnosis decision on spindle status.

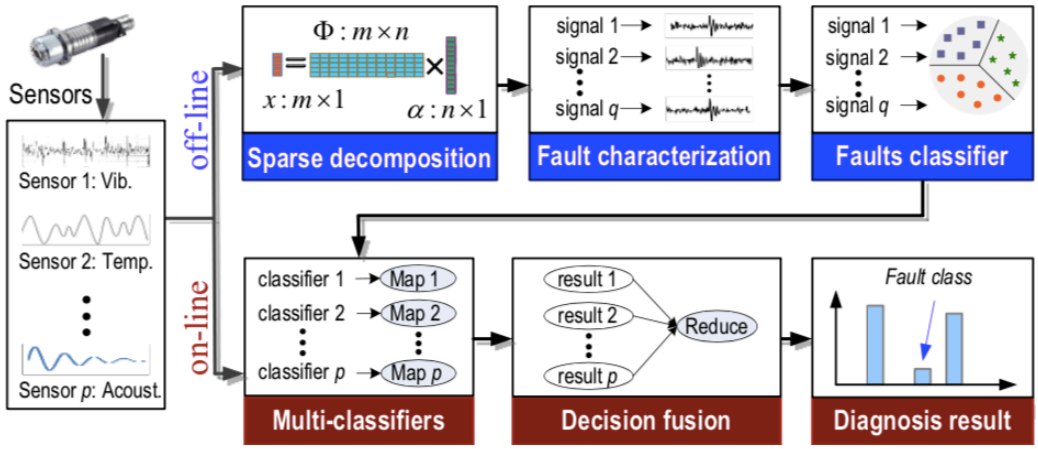


Figure 4. Sparse representation-based model for diagnosis

2.4.4 Task D: Prognosis of machine performance degradation and prediction of remaining useful life (RUL) based on particle filter

Current Problem: Prognosis, as a complementary task to diagnosis, plays a critical role in spindle system performance tracking and remaining useful life (RUL) prediction. Currently, techniques for prognostic modelling can be classified into two categories: data-driven and model-based, depending on the availability of physical knowledge about the system. Data-driven techniques, which are typically implemented by neural networks, support vector machine, or other machine learning techniques, establish black-box models to characterize the relationships between system states and measurements. It is assumed in this process that the evolution of the system states would exactly follow the pattern inherent to the historical data, which however may not be realistic and accurate, due to the nonlinear relationship between the two in many dynamic systems. Model-based approach, in comparison, builds grey-box models based on partial physical or empirical knowledge. This approach, mostly achieved by Bayesian inference, employs a filtering method to account for the stochasticity of the process and noise embedded in the measurement, providing more meaningful and comprehensive results as compared to a purely data-driven approach. However, the up-to-date prognosis technologies still suffer from several constraints:

- Limited tracking capability: Most prognosis methods are limited in tracking system degradation with varying rates or transient changes caused by sudden occurrence of faults. In addition, diagnosis results such as the time and severity of fault occurrence, cannot be incorporated into prognosis model.
- Limited adaptability: In general, a prognosis model based upon specified machine and measurement data is not guaranteed to be directly applicable to the prognosis of other machines or even the same type of machines in a different operation environment. This is because factors such as operating conditions and maintenance actions, which affect system performance degradation, are typically not accounted for in the prognosis model due to difficulty in the modeling.
- Incomplete evaluation of uncertainty: Current prognosis methods only quantify uncertainty associated with sensor measurements. It is however important to also evaluate uncertainty from modelling errors due to: 1) assumptions and simplifications made and/or incomplete training; 2) nonlinear relationship between measurements and system states; and 3) randomness associated with future degradation because of new failure occurrence and changes in the operational conditions.

Developed Solution: To address the above challenges, a hybrid prognosis model taking advantage of both

data-driven and model-based approaches has been developed in this research, with the structure shown in Fig. 5. Consider crack growth prediction as an example. When physical knowledge or experiential information is available, an analytical model is established using physical information and then updated through estimation of unknown material parameters in the model, following a model-based approach such as particle filter. A common drawback is that the model based upon physical knowledge (such as Paris' law) doesn't involve machine settings and maintenance actions as parameters, because it is difficult to explicitly describe them in physics or analytically. To compensate for the limitation and construct a more comprehensive and robust prognosis model, a data-driven approach is taken to estimate the relationship between these factors and the spindle health state. To match prediction results (probability distribution) from the model-based approach, the relationships estimated by a data-driven approach are first characterized as a certain probability distribution. The parameters are then estimated by machine learning methods, such as support vector regression. As a result, spindle system degradation is described by multiple probability distributions containing different factors that affect degradation process. Finally, model fusion of probability distributions is performed to obtain a comprehensive prognosis model.

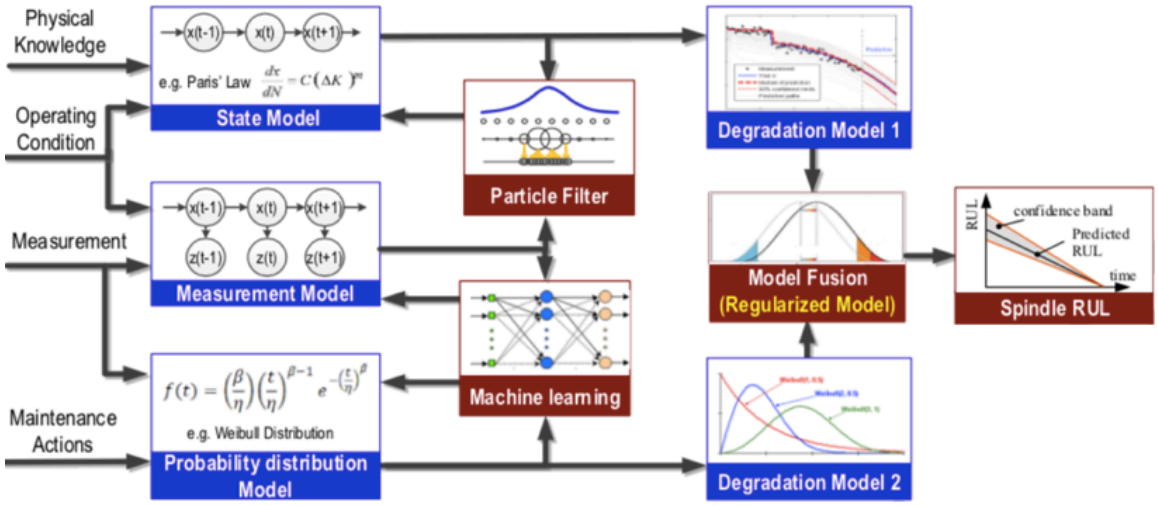


Figure 5. Regularized prognostics model

3. KPI's & Metrics

The project provides a generic framework for hybrid cloud-based machine and process monitoring, diagnosis, and prognosis and a prototype that can be integrated into legacy machines and general purpose CNC machines by both SMEs and large OEMs. The final project deliverables include:

- An interoperable data acquisition and on-premise cloud computing platform providing scalable data collection and processing for hundreds of manufacturing machines on factory floors;
- A public cloud platform integrated with on-premise private cloud for processing real-time data streams, executing parallel machine learning algorithms, generating big data analytics, and visualizing data;
- A set of experimentally tested algorithms enabling data-driven intelligence for online machine fault diagnosis and prognosis in various types of manufacturing machines and processes, executable on a hybrid cloud computing platform.

As with any project, a set of monthly, quarterly, and annual reports were provided. At the end of the project, a final technical report (this report) was provided. Each report contained high level technical status, project risks and opportunities, schedule status and/or schedule modifications, project issues, budget expenditure,

and cost share. Briefly these reports were:

- Monthly technical reports
- Quarterly technical and financial reports
- Annual technical report
- Final technical report

Metric	Baseline	Goal	Results	Validation Method
Data-driven methods	Fuzzy theory, neural network, wiener process, and gamma process	Parallel machine learning and data mining approaches	Cloud-based parallel machine learning algorithms were developed.	Use case
Software portability	Lack of portability due to incompatibility between applications and computing systems	Improved portability enabled by container technology	A plug-and-play, interoperable data acquisition system was developed.	Use case
Computing scalability	Limited scalability by adding or removing computing resources	High scalability enabled by cloud computing	An Azure-based high performance cloud platform was developed.	Use case
Data accessibility	Limited access to data due to the lack of data synchronization	Ubiquitous access to data enabled by centralized cloud storage	A container-based scalable private cloud was developed.	Use case
Data volume	Limited data storage	Potentially unlimited and scalable data storage	A container-based scalable private cloud was developed.	Use case
Infrastructure flexibility	In-house ICT infrastructure and/or private cloud	Integration of both private and public cloud in flexible hybrid cloud model	An Azure-based high performance cloud platform was developed.	Use case
Security and cost-effectiveness	Private clouds are the most secure but also most expensive; Public clouds are the least secure but least expensive.	Hybrid clouds offer a reasonable level of security while providing the most powerful and least expensive computing resources.	Low development cost and total cost of ownership	Use case

4. Technology Outcomes

4.1 Task A: Developing a standardized sensing system and a private cloud for collecting and processing data

Task A.1: Interoperable data acquisition system. Task A.1 focuses on developing a standardized interoperable data acquisition system that can connect hundreds of manufacturing machines on factory floors. For legacy machines without sensors, an interoperable and modular "drop-in" sensing system is developed for real-time data collection. For general purpose CNC machines with sensors and fieldbuses, a selection of protocol adapters is developed. These adapters are pre-installed on a gateway device, connecting CNC machines with different fieldbuses.

Task A.2: Scalable computing platform for machine intelligence. Task A.2 focuses on bringing computing power and intelligence to manufacturing machines by introducing light-weight multi-core computing

devices and Linux container-based computing software framework to factory floors. The goal is to develop an on-premises computing platform capable of executing online diagnostic and prognostic models in the form of Linux containers.

The goals and objectives for this research is to develop (1) an interoperable data acquisition system and (2) a scalable computing platform for machine intelligence. Below are some of the key characteristics for the system to be successfully adopted:

- Open and accessible – leverage open-source SW and COTS as much as possible
- Low Cost – low development cost and total cost of ownership
- Plug-and-Play – enable "drop-in" for data collection, analytics, and more
- Fault tolerant – high availability and high assurance
- Extensible – ease of use for "app" development
- Scalable – can easily scale up and down for resource management
- Ease of setup – easy to setup and maintain

We adopted the modular design approach in designing the entire system. Fig. 6 shows the system architecture of such a modular platform that enables data-driven smart manufacturing.

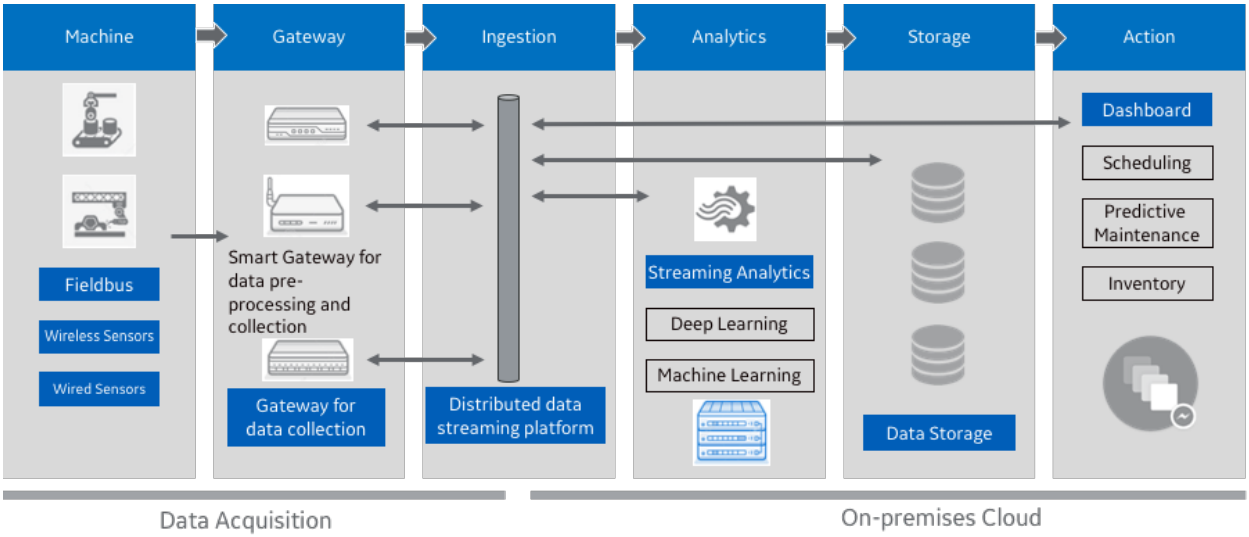


Figure 6. System architecture: a modular platform enabling data-driven smart manufacturing.

Specifically, the platform includes six modules from machine data connectivity to data analytics and user interface.

- Machine Modules provide protocol adapters between the platform and manufacturing machines. The adapters collect data from machines with MTConnect protocols and wireless and wired sensor nodes.
- Gateway Modules are embedded computers that host protocol adapter software that receive data from machines. They also provide optional pre-processing and data aggregation functionalities before sending the data to on-premises cloud through ingestion module.
- Ingestion Modules provide the internal distributed data streaming of the platform. Kafka is chosen as the vehicle due to its maturity and performance. Data from gateway module is published to the Kafka data broker, while Analytics, Storage, and other modules subscribe to specific topics of data from the broker and receive the data once it is available.
- Analytics Modules subscribes to the data streaming bus, and performs respective analytics once data is available.

- Storage Modules also subscribe to the data streaming bus, and store the data into a time-series database.
- Actions Modules also subscribe to data streaming bus and provide data visualization, scheduling, predictive maintenance, and cloud gateway adapter.

Given the complexity of the operating environment and available infrastructure in a factory floor, coupled with the status quo of the computing devices and the variety of programing language the community prefer, there are quite a few challenges in designing such a system.

- Homogeneous architecture – From data acquisition to cloud analytics to user dashboard, we are designing this architecture for a variety of device types and use cases. A unified architecture that creates homogeneous environment for software development and deployment is critical to the success.
- Constrained hardware resources - Many devices may lack computing and memory resources. Their ability to run applications and process data is therefore limited. An architecture that can scale up and down easily to fit in the targeted devices is desired.
- High volume data throughput – Manufacturing data can be overwhelmingly vast. Collecting and storing them in a distributed environment is difficult. A high performance, high throughput messaging bus that can reliably deal with high volume data is desired.
- Geographic distribution - In many use cases, machines are spread across a large geographic area in a factory or even across sites. Collecting data from and delivering software to them is very challenging even with high performance messaging bus. A distributed architecture that can deploy processing close to the data is critical to reduce traffic and speed up decision making.
- Multiple CPU architecture – From sensor to server, devices may have different CPU architecture such as ARMv8, ARMhf, AMD64 and i386. Software framework that can easily cross-compile and deployable on different CPU architecture is needed.
- Polymorphism – In order to be truly plug-and-play, the platform needs to be able to host applications that is written in different programming languages – Java, Python, C/C++, golang and so on. The service API's needs to be standardized to be language agonistic.

These challenges drove us to adopt a maturing technology called Linux Containers. Containers technology has become the mainstream virtualization framework for the Cloud in the last couple of years. We modified it to fit embedded devices in our application. All the modules within the platform leverages Docker Container technology, as shown in Fig. 7, enabling the plug-and-play design. Each module lives inside a Docker Container Image, which could be written in various programming languages, e.g., Java, Python, etc., and can be started or stopped as needed by the end user of the platform. For example, a user could start a data analytics once the machine data is available through the Kafka-based data streaming bus, and perform analysis accordingly. More details on Docker Container are included in later sections.

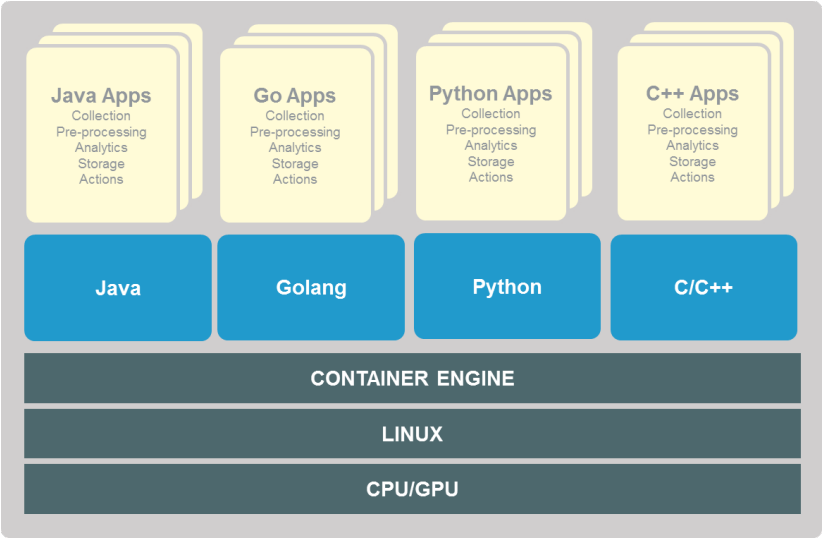


Figure 7. Docker container based plug-and-play design.

Wireless Sensor Nodes

The Bluetooth low energy (BLE) SensorTag from Texas Instruments (TI) is chosen in this project to demonstrate data acquisition via wireless connectivity. There are many wireless protocols available: WiFi, ZigBee, Bluetooth, etc. BLE is preferred due to its low-power consumption feature, which is essential for long-term sensor deployment and machine monitoring. In addition, BLE also has easy-to-configure feature, and is widely adopted in both consumer and industrial applications.

BLE is sometimes referred to as "Bluetooth Smart", which is a light-weight subset of classic Bluetooth. BLE was introduced as part of the Bluetooth 4.0 core specification. Generic Access Profile (GAP) is critical for BLE, since it controls connections and advertising in Bluetooth. GAP is what makes BLE device visible to the outside world, and determines how two devices can interact with each other. GAP defines various roles for devices, but the two key concepts to keep in mind are Central devices and Peripheral devices.

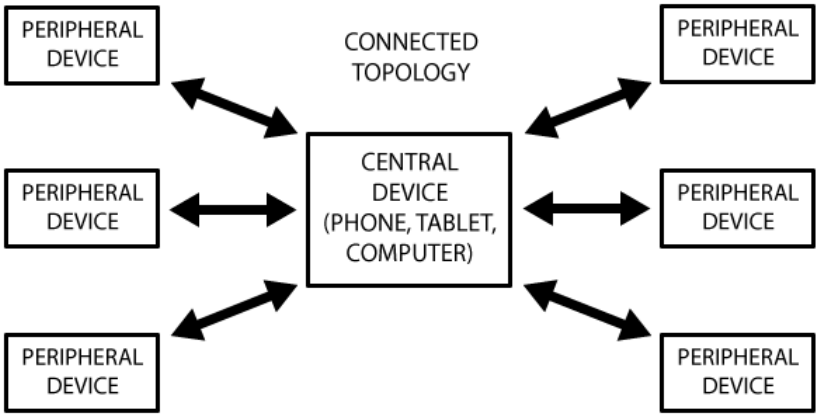


Figure 8. BLE Central and Peripheral topology

Peripheral devices are small, low power, resource constrained devices that can connect to a more powerful central device. In this project, the Peripheral devices are the SensorTag devices that can be deployed to monitor both environmental conditions and machine conditions. Central devices can be the mobile phone or tablet that one connects to with far more processing power and memory. We choose to use Intel NUC computer as our central device in this project. Note that one central device can be connected with multiple peripheral devices. The overall architecture and topology is shown in Fig. 8.

Another important concept in BLE connectivity is Generic Attribute Profile (GATT), which defines the way that two BLE devices transfer data back and forth using concepts called Services and Characteristics. It uses a generic data protocol called the Attribute Protocol (ATT), which is used to store Services, Characteristics and related data in a simple lookup table using 16-bit IDs for each entry in the table. GATT comes into play once a dedicated connection is established between two devices. Note that BLE connections are exclusive. That is, a BLE peripheral can only be connected to one central device at a time. As soon as a peripheral connects to a central device, it stops advertising itself and other devices no longer be able to see it or connect to it until the existing connection is broken. Establishing a connection is also the only way to allow two-way communication, where the central device can send meaningful data to the peripheral and vice versa.

For BLE data acquisition, two important concepts are services and characteristics. Services are used to break data up into logic entities, and contain specific chunks of data called characteristics. A service can have one or more characteristics, and each service distinguishes itself from other services by means of a unique numeric ID called a UUID, which can be either 16-bit (for officially adopted BLE Services) or 128-bit. Characteristics are the main point that users interact with their BLE peripherals. They are also used to send data back to the BLE peripheral, since users are also able to write to characteristic. In this project, we develop software to collect data from various sensors using different characteristics, i.e., UUIDs, which is described in detail in next section.

In addition to the BLE CC2540/2541 SOC chip, the Bluetooth SensorTag has the following sensors: Contactless IR temperature sensor (TI TMP006), Humidity Sensor (Sensirion SHT21), Gyroscope (Invensense IMU-3000), Accelerometer (Kionix KXTJ9), Magnetometer (Freescale MAG3110), Barometric pressure sensor (Epcos T5400), On-chip temperature sensor (Built into the CC2541), Battery/voltage sensor (Built into the CC2541). A picture of the SensorTag with various sensors are shown in Fig. 9. In this project, we collect data from five sensors: temperature, movement, humidity, barometric pressure and optical sensors.

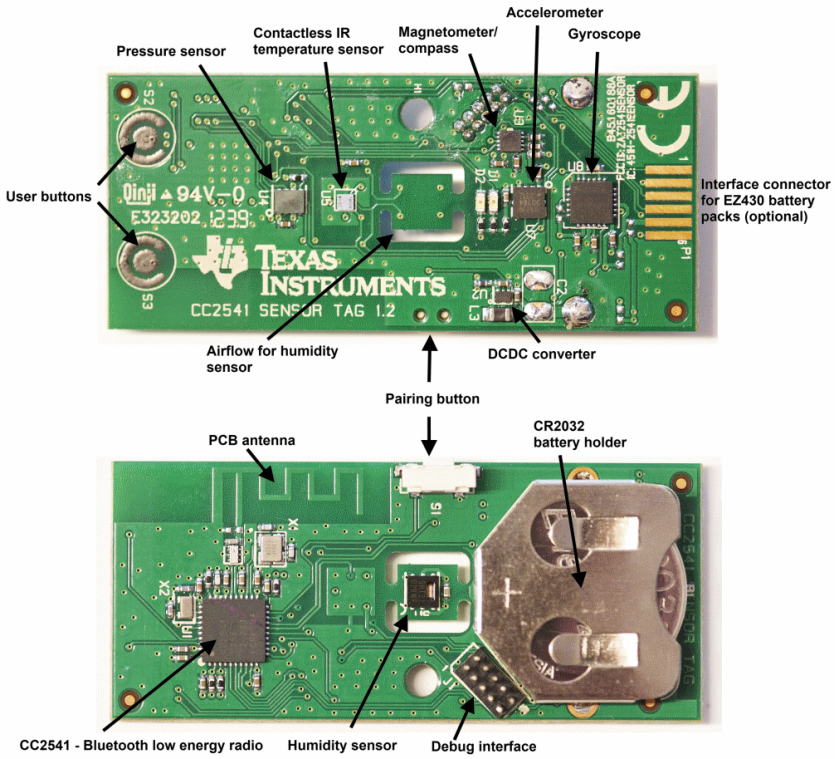


Figure 9. TI SensorTag

Data acquisition software was developed with Golang. The software features discovering, connecting and acquiring data from a BLE device. We use Golang and open source BLE GATT library to implement those functionalities. The software was developed and deployed as Docker image.

Wired Sensor Nodes

To meet one of GE's business requirements, accelerometers 602D01 and 607A11 from PCB Piezotronics are chosen as the sensors to monitor machine vibration. Both sensors have a sensitivity of 100 mV/g, that is 10.2 mV/(m/s²). However, the frequency ranges are different: 602D01 has a stable frequency response within the range of 0.5 to 8 kHz; while the range for 607A11 is from 0.5 to 10 kHz. In addition, accelerometer 607A11 has 30-ft integral cable with swiveled base, while 602D01 has configurable and detachable cable.

A critical aspect of collecting high-quality vibration signal is the sensor mounting method. Magnetic mounting provides a convenient way of making portable measurements. It is commonly used for machinery monitoring application. Two magnet mounting studs are chosen to work with the accelerometers mentioned above. One is flat surface magnet stud and the other is curved surface magnet to provide more flexibility for deployment.

Finally, both sensors require a constant current 18 to 30 VDC power source for proper operation. A signal conditioner is used to provide well-regulated DC power in addition to signal conditioning function, which is discussed next.

To guarantee the quality of collected vibration signal, a four-channel sensor signal conditioner 482C16 is chosen to work with the accelerometers. The signal conditioner has 12-bit accuracy signal conditioning for up to four sensors with BNC connectors. It has programmable gain that can be adjusted incrementally from x0.1 to x200. One of National Instrument's data acquisition card was chosen for sampling and acquiring the data. Using the C API released in NI-DAQmx Base 15, the acquisition software was developed and deployed as Docker image.

MTConnect Protocol Adapter

Based on one of GE's business requirements, a MTConnect protocol adapter was developed to collect data from GE's manufacturing machines. MTConnect is a standard that defines how manufacturing machines can provide structured and contextualized data. Machines equipped with MTConnect means that they can provide data in standard XML format with data item definitions that do not vary by manufacturer. The protocol adapter has been developed using a Python HTTP client and deployed on the gateway device in the form of a Docker image.

Gateway Device

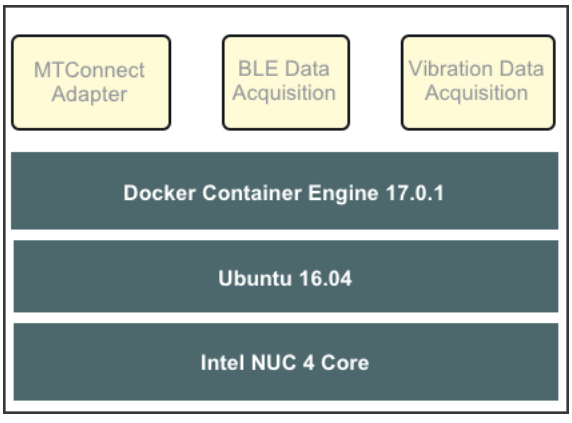


Figure 10. Gateway software stack

Gateway are embedded computers that are physically connected to manufacturing machines (wired or wireless) and collect data through protocol adapters or data acquisition software. Since the framework is Linux container based so theoretically any modern embedded computer with Linux OS can serve as a gateway. In our experiment, we used two types of embedded computers: Raspberry Pi and Intel NUC. Fig. 10 shows the software stack for an Intel NUC gateway.

Real-time Data Streaming

In addition to sensor data acquisition, we use Apache Kafka to implement real-time data streaming. Kafka is chosen in this project, since it is well suited for building real-time streaming data pipelines that reliably collect data between applications or sub-systems. In addition, a data streaming platform such as Kafka makes the on-premises cloud highly scalable, elastic, distributed, and fault-tolerant. Open source data analytics package Spark can also be easily integrated with Kafka. Apache Kafka is an open source distributed streaming platform with three key capabilities:

- Users can publish and subscribe to streams of records. It is similar to a message queue system.
- Users can store streams of records in a fault-tolerant way.
- Users can process streams of records as they occur.

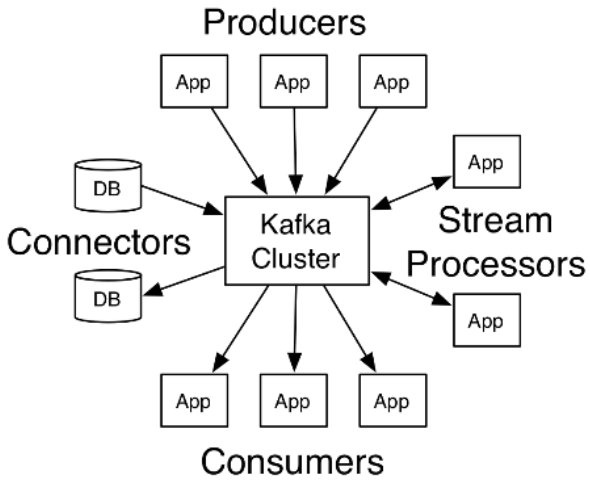


Figure 11. Kafka structure

Kafka has four core APIs: Producers, Consumers, Connectors and Stream Processors. In this project, we mainly use the Producer and Consumer APIs.

- Kafka producer - The Producer API allows an application or sub-system to publish a stream of records to one or more Kafka topics. Topics in Kafka are always multi-subscriber. For each topic, the Kafka cluster maintains a partitioned log. Each partition is an ordered, immutable sequence of records. The Kafka producer is responsible for choosing which record to assign to which partition within the topic.
- Kafka consumer - The Consumer API allows an application to subscribe to one or more topics and process the stream of records. The Kafka consumers label themselves with a consumer group name and reach record published to a topic is delivered to one consumer instance within each subscribing consumer group.

Although Kafka itself is written in Scala and Java, Kafka has many language bindings, such as C/C++, Golang, Python, etc. In this program, we use Kafka docker image together with Zookeeper to deploy and maintain cluster of Kafka brokers in the on-premises cloud. Zookeeper is distributed systems configuration management tool that provides features for distributed applications like distributed configuration

management, leader election, consensus handling, coordination and lock. The deployment was tested on a multi-core server.

On-premises Cloud Computing Platform

Traditionally, setting up and managing on-premises cloud platform for data center and stream computing was difficult. Virtualization technology such as VMware and Openstack were the only things available. Such solutions are expensive and inflexible in nature and can hinder the adoption of the system on a factory floor. In this program, we explore the use of Linux container technology, specifically Docker, for the foundation of the on-premises cloud. Recent years, with the explosion in popularity of Docker containers, running on-premises is becoming more flexible, and cost-effective.

Docker is a tool to make it easier to create, deploy, and run applications by using Linux containers. Containers allow developers to put their software application together with all of its dependencies in one package. By doing so, the application can be shipped and deployed easily and run on any other Linux machine regardless of any customized settings. Docker is very much like a traditional virtual machine. However, rather than creating a whole operating system like traditional virtual machine does, Docker containers share same Linux kernel and file system. This gives a significant performance boost and reduces the size of the application.

In addition to Docker engine itself, we also leverage Docker Compose for on-premises cloud management. Compose is a tool for defining and running multi-container applications. Using YAML files, one can configure the application's services, and then start all the services from the configuration with a single command. Fig. 12 shows an example of the on-premises cloud setup used in this program.

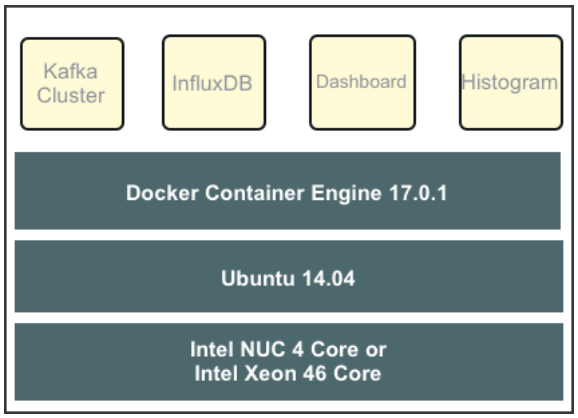


Figure 12. On-premises cloud server software stack

Docker engine 17.0.1 was installed on a multi-core server and a multi-container application was deployed using Docker Compose. The application hosts a Kafka broker cluster for gateway devices to stream machine data, a database writer subscribes to the data stream and store them in InfluxDB database, an exemplary analytics container conducts histogram analysis to the data and a dashboard container retrieves data from database and visualize them for decision making on the factory floor.

4.2 Task B: Integrating a public HPC cloud with a private cloud for big data analytics and visualization

Task B includes three subtasks: (1) integrating the private cloud with the Microsoft Azure public cloud, (2) implementing cloud-based machine learning, big data analytics, and visualization tools and services provided by Microsoft Azure, (3) testing the Microsoft Azure cloud of the hybrid cloud prototype using real-time data streams. The relationship in this project between private cloud and public cloud is listed in Figure 13.

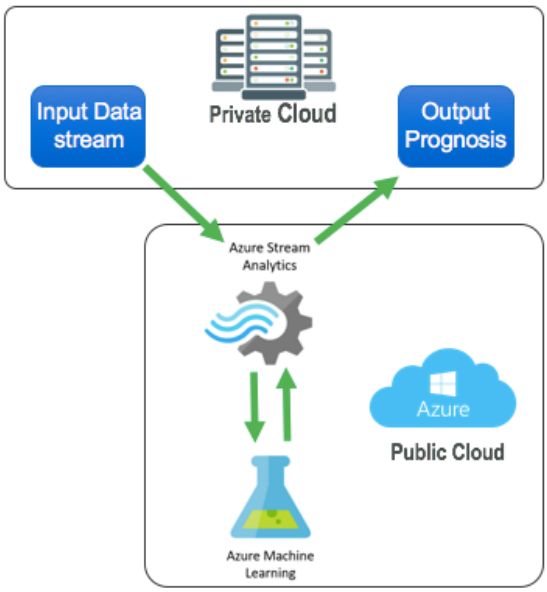


Figure 13. The relationship between private cloud and public cloud

The input data stream is being collected in private cloud and being transmitted to Microsoft azure public cloud. Transmitted data stream are being processed in public cloud for system diagnosis and prognosis. The final step is to transmit output results from public cloud to private cloud for output prognosis. The public cloud is Microsoft Azure; the function of Microsoft Azure is mainly realized by Azure Stream Analytics. Figure 14 shows the function of Azure Stream Analytics.

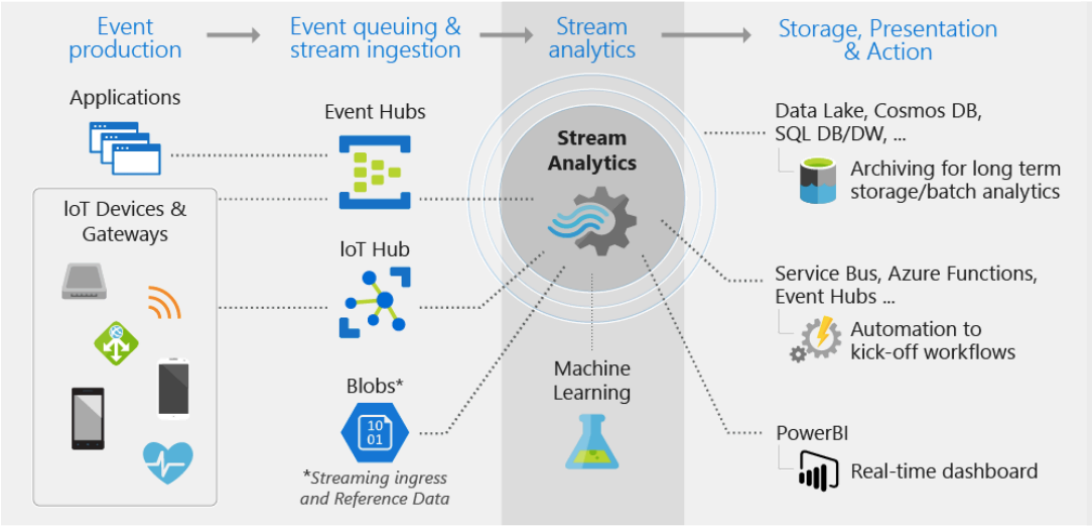


Figure 14. Several functions of Azure Stream Analytics

Azure Stream Analytics seamlessly integrates with Azure IoT Hub and Azure IoT Suite to enable powerful real-time analytics on data from your IoT devices and applications. Additionally, Azure Stream Analytics is available on Azure IoT Edge. Azure Stream Analytics on IoT Edge empowers developers to deploy near-real-time analytical intelligence closer to IoT devices so that they can unlock the full value of device-generated data.

In this project, Blob storage and Azure machine learning studio are used for cloud-enabled diagnosis and prognosis. Blob storage is Microsoft's object storage solution for the cloud, which could store streaming sensor data, store data for analysis by an on-premises or Azure-hosted service. Figures 15-16 show the creation of Blob storage and the connection between Blob storage and Azure stream analytics.

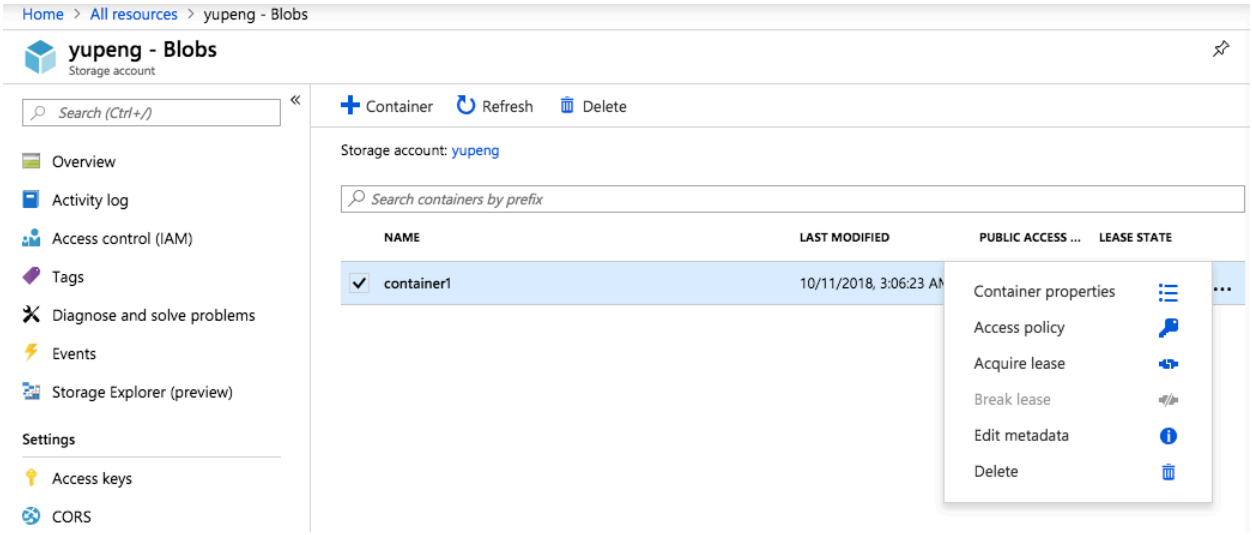


Figure 15. An example of Blob storage's creation

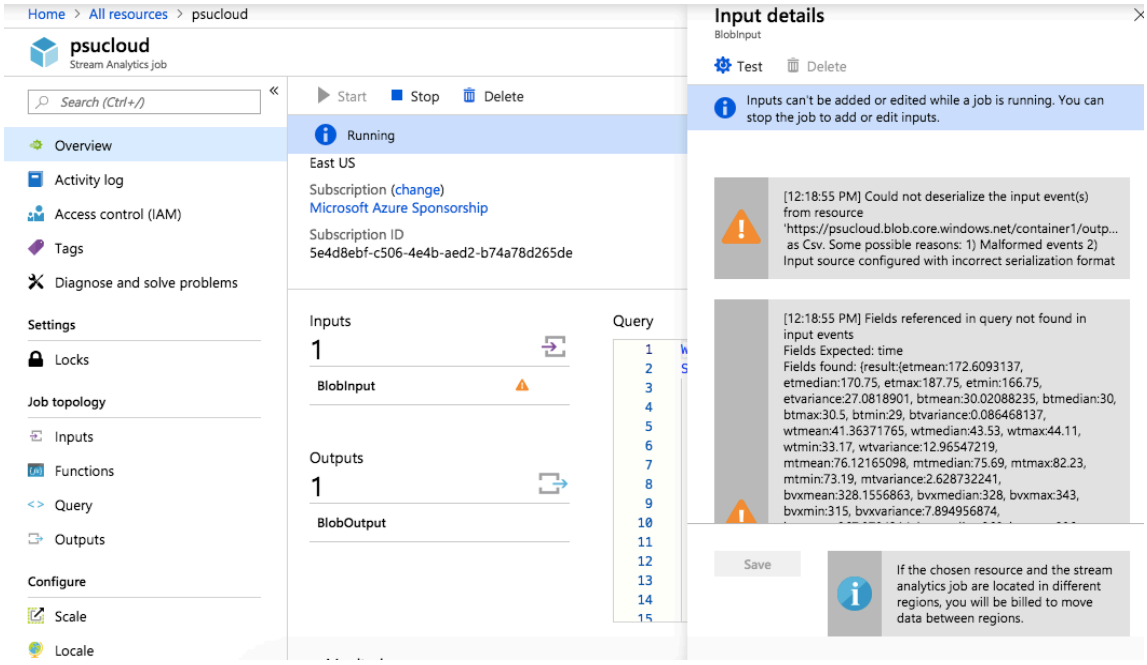


Figure 16. Blob storage connects with Azure stream analytics

After streaming data is uploaded from private cloud to public cloud, sensor data is being processing with Machine learning studio in the Azure stream analytics. Microsoft Azure Machine Learning Studio is a collaborative, drag-and-drop tool you can use to build, test, and deploy predictive analytics solutions on your data. Machine Learning Studio publishes models as web services that can easily be consumed by

custom apps or BI tools such as Excel. Figure 17 shows the configuration of machine learning studio in the Azure stream analytics.

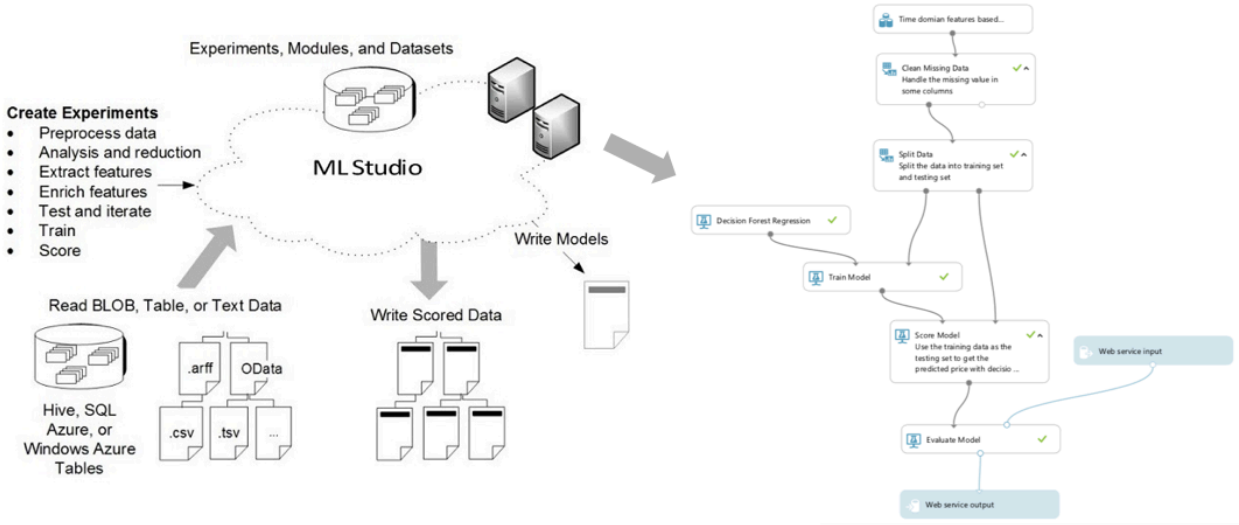


Figure 17. A configuration of machine learning studio in the azure stream analytics

To perform data-driven diagnosis and prognosis methods on cloud, training and testing phases are deployed on web service. Figure 18 show the training and testing phases on the web service.

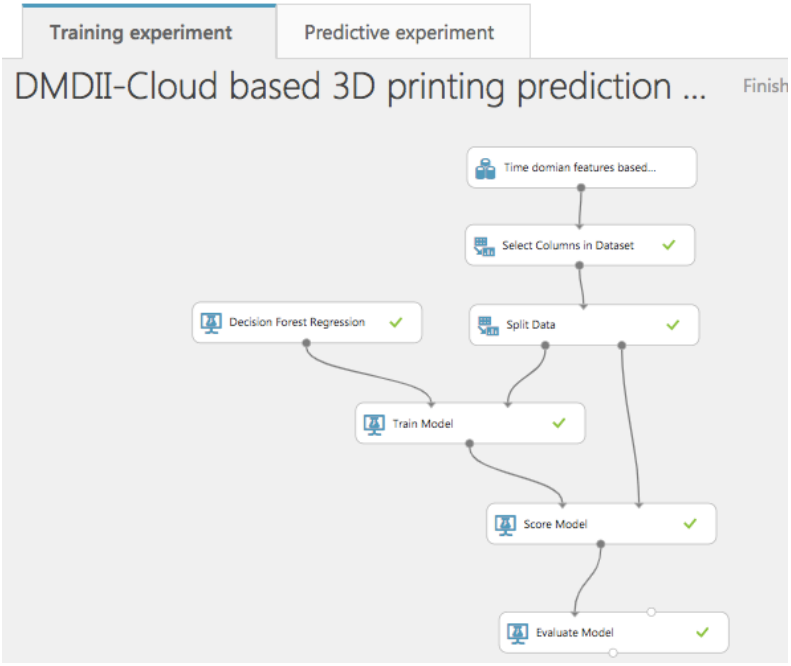


Figure 18. Training and testing process on the web service of machine learning studio

To transmit diagnosis and prognosis output from public cloud to private cloud, Blob storage is utilized. The way to setup Blob storage is similar as previous setup procedure. Finally, SQL code is utilized to connect all single units. Figure 19 shows the connection all units by using SQL.

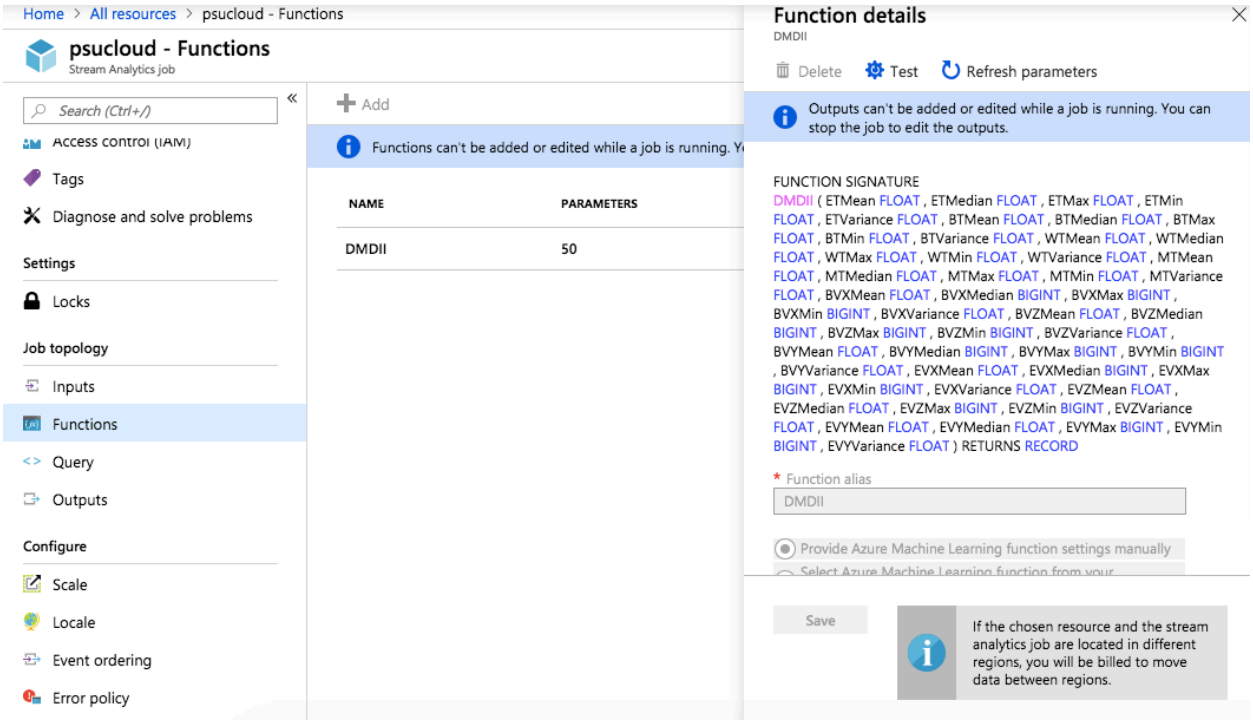


Figure 19. SQL code for the connection between private cloud and public cloud

4.3 Task C: Diagnosis of machine structural faults based on sparse representation and dictionary learning

Task C includes two subtasks: 1) dictionary learning for fault characterization and 2) nonlinear sparse multi-faults classifier.

Analysis of sensing data provides the technological basis for fault diagnosis of many manufacturing equipment. Choosing an effective data model is essential for data pattern recognition and subsequently, better fault recognition. Commonly seen sensing signal, such as vibration, is *dense* in its raw form in time domain and is difficult to deal with directly. To address this limitation, signals have been represented by a *sparse* model for the first subtask in this project. The basic concept of the sparse model is to transform a signal into a linear combination of atoms of a dictionary for which the number of the atoms used to represent the signal is minimized, e.g., a sparse representation. Sparse representation is computed using greedy method, which iteratively generates a sorted list of atom indices and weighting coefficients until the representation error is minimized for the given signal. The dictionary itself can then be regarded as a reflection of the sparse pattern of the original signal, which can then be leveraged for condition-related pattern discovery [1].

As the signals are subject to noise and variation, a set of signals for a given machine condition is often utilized for condition-related pattern discovery to account for these factors. Traditional sparse representation uses empirical, fixed dictionary, which limits the capability of the dictionary to adapt to the underlying, condition-related pattern of each machine condition, as reflected by the overall uncontrolled representation error of the set of signals. To address this limitation, in this project, dictionary learning has been integrated to iteratively reduce the overall representation error and obtain the optimal dictionary for the given machine health condition. Specifically, sparse representation is alternatively performed along with dictionary update. In each update step, the residual error associated with one individual atom is computed, indicating the representation error without this specific atom. Then this atom is replaced with a new atom that maximizes the residual error reduction. The same update step is carried out over all atoms

to ensure the maximization of the residual error reduction associated with the whole dictionary. The next iteration of sparse representation and dictionary update then follows, and the process continues until an optimal dictionary is obtained, quantified by the minimization of the overall representation error of all signals [7].

A contribution to dictionary learning achieved in this project is that it does not impose constraints on the atoms (unlike the discrete Fourier or wavelet transform in which the atoms are required to be orthogonal to each other), and therefore providing a greater adaptability in signal modelling and facilitate the capture of condition-related signal patterns through the iterative dictionary update steps. For the task of machine multi-fault diagnosis, individual dictionary is learned for each individual machine health condition as shown in the Fig. 20.

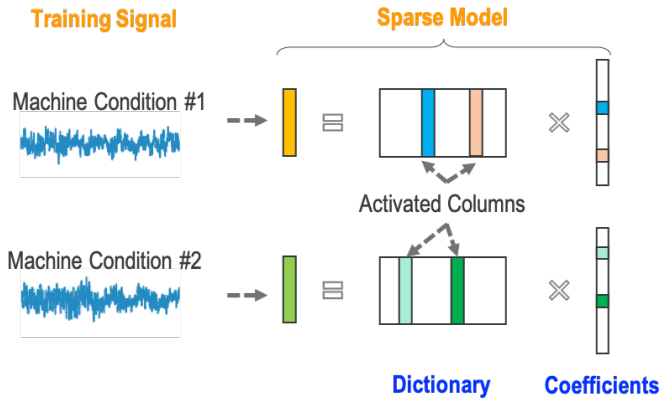


Figure 20. Dictionary learning for machine sensing signal representation

For the second subtask, to better adapt to the non-linearity embedded in the signals, kernel method has been investigated and improved. The method maps the signals from low-dimensional input space into high dimensional feature space for non-linearity handling. Traditional kernel method applies "kernel trick" at algorithm level, which takes advantage of the inner-products within the algorithm to avoid explicit computation of the signals in feature space, as these inner-products can be computed explicitly using the signals in input space through kernel function, and they are represented by a kernel matrix [1].

The limitation of the "kernel trick" is two-fold: 1) it is algorithm-dependent. Therefore, ad-hoc method has to be developed for each algorithm to re-formulate it into inner-product operations for the kernel method to work and 2) it does not apply to the algorithm that does not involve inner-product operations. Dictionary learning is one such example. The dictionary update step does not involve inner-product operations, as the atoms are updated one-by-one. Therefore, the "kernel trick" does not apply. To overcome this issue, a data-level kernel method has been developed in this project to address this limitation.

The basic concept is to construct virtual samples that numerically represent the signals in feature space by leveraging the property of the kernel matrix. The semi-definiteness of the kernel matrix means its eigen-decomposition exists and the matrix can be expressed symmetrically as the inner-product of a set of samples with itself. Therefore, this set of samples numerically shares the same property as the corresponding set of signals in the feature space and can be utilized as "virtual samples" [7]. The conversion of signals into virtual samples occur at data level and it is independent of the algorithm used. Therefore, no ad-hoc re-formulation is required, and dictionary learning can be directly carried out with the virtual samples without any modification, as illustrated in Fig. 21.

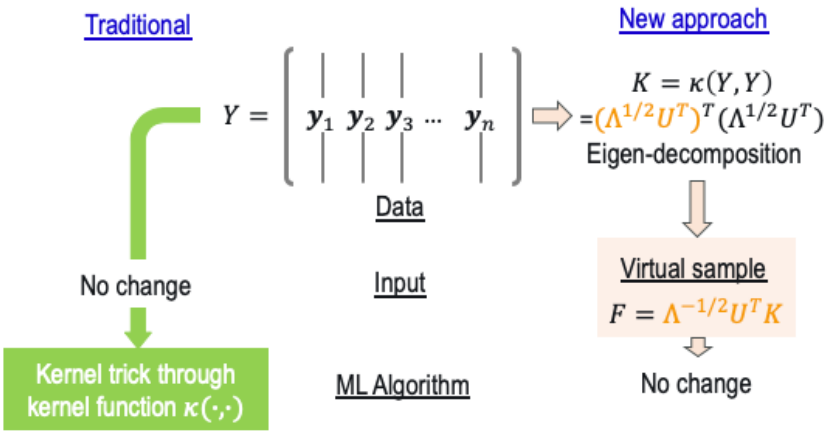


Figure 21. Comparison of the traditional algorithm-level and developed data-level kernel method

For multi-fault machine diagnosis, a sparse multi-fault classifier has been constructed with the dictionaries learned from sensing signals of different machine condition. For a testing sample from one of the machine health conditions, sparse representation is carried out over the main dictionary, which is formed by concatenating all dictionaries. The sample representation error is evaluated using each dictionary with the corresponding representation coefficients. The final classification of the testing sample is the class whose dictionary and corresponding coefficients produce the smallest representation error [1,4,7], as illustrated in Fig. 22. The sparse multi-fault classifier has two desired properties: 1) it can be readily expanded to account for new machine conditions by simple concatenation of the new dictionaries learned from these new conditions and therefore, is suitable for the changing manufacturing settings, and 2) learning of dictionary is independent from each other, therefore, it is ideal for the execution in a parallel computing platform which can improve the computational efficiency.

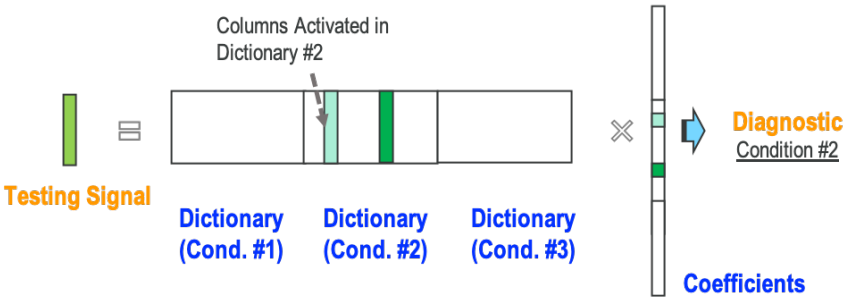


Figure 22. Sparse multi-fault classifier

4.4 Task D: Prognosis of machine performance degradation and prediction of remaining useful life (RUL) based on particle filter

This task includes two subtasks: 1) advanced degradation tracking under varying rates and transient changes through advanced particle filter-enabled stochastic modeling; and 2) development of a regularized prognosis model for universal application by combining data-driven modeling techniques for characterization of system performance and stochastic modeling for tracking the system performance degradation and predicting the remaining useful life (RUL).

For the first subtask on tracking time-varying system performance degradation, improvements on particle filter that have been realized through this project include: 1) a local search particle filter (LSPF) has been developed through a perturbation analysis, to have the particles dynamically follow the variation of a posterior probability density function (PDF) to be estimated (i.e. a degradation with time-varying

degradation rates), with the convergence property of the LSPF mathematically proved; 2) a multi-mode LSPF has been developed for tracking and predicting a system performance degradation with time-varying modes (e.g. from linear degradation mode to exponential degradation mode); and 3) a total variation filter has been integrated with the LSPF to detect transient performance changes and improve the accuracy in system performance degradation and RUL prognosis [2,10].

Specifically, the first improvement is to solve a problem inherent to PF is *particle degeneracy*, which refers to the phenomenon where weights of most particles become negligible after several iterations. This is mainly caused by a poor initial guess of the prior PDF, from which the particles being generated are not effective in searching the space where the posterior PDF spans. The problem worsens when the posterior PDF is time-varying. In this project, a resampling technique was developed to adaptively relocate particles according to their performance in the last iteration. This was realized by adding perturbations to the particles. The perturbation for each particle is sampled from a normal distribution, which is determined by the particle's estimation accuracy in the previous iteration step. A particle is assigned with a small perturbation to tightly explore the area near the peak(s) of the posterior PDF to be estimated, if the particle's estimation is close to the optimal estimation. Otherwise, the particle is assigned with a larger perturbation. Relocating particles in such a fashion not only increases the particle diversity, but also enables particles to move with the variation of system state and parameters to track time-varying systems. A shrinkage coefficient is involved in generating the perturbation to ensure that the resampled particles would gradually converge to the optimal location, to narrow down the confidence interval associated with the estimation and prediction. In other words, the shrinkage coefficient enables that the variance of the particles gradually decreases, if the posterior PDF to be estimated is fixed. Through a mathematically rigorous derivation, a proper value of the shrinkage coefficient can be recursively obtained to ensure the convergence of the resampled particles [2]. Figure 23 shows the illustration of the adaptive resampling process.

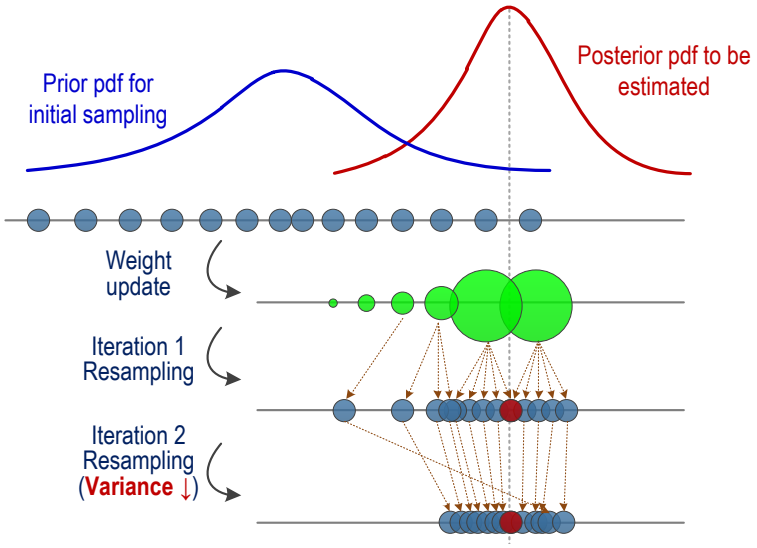


Figure 23. Local search particle filter with adaptive resampling strategy

To track system performance degradation with time-varying degradation modes (governed by different degradation functions), a method for tracking jump Markov non-linear system (JMNLS) has been investigated. To track a JMNLS, a set of linear or non-linear degradation modes are predefined according to the physical/empirical knowledge or statistical analysis of historical data, with each mode corresponding to an individual vibration variation (representing performance degradation) scenario. In this project, a multi-mode PF has been developed for tracking time-varying system degradation, based on a generalized linear degradation mode and a generalized exponential mode. Next, a finite-state Markov chain switches between the two modes, reflecting the variation in degradation patterns. The mode switch is automatically performed

at each iteration, by calculating the likelihood of sensor measurement given each mode and deciding which mode better describes the current degradation scenario [2,10]. Figure 24 illustrates the two-mode PF for system estimation and mode transition.

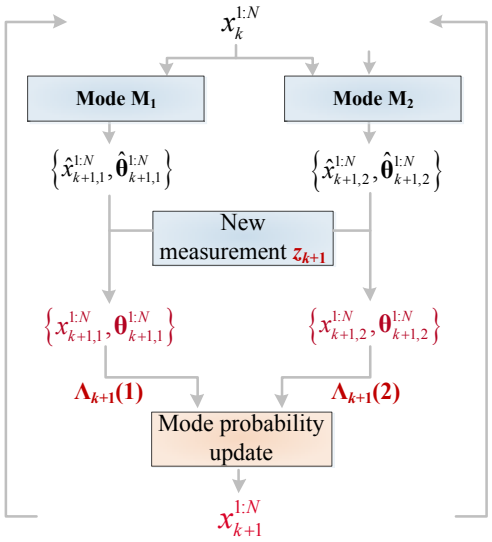


Figure 24. Multi-mode particle filter

During the system performance degradation process, abrupt faults may occur and lead to transient performance drops. Particle filter is not able to detect the transient performance drops. To address the challenge, a total variation (TV) filter has been investigated and integrated with PF in this project to improve the capability of PF for system performance tracking and prediction. In such a framework, performance degradation is divided into gradual degradation (estimated by LSPF) and transient performance drops (detected by TV). While LSPF performs a step-by-step estimation upon each arrival of new measurement, the TV filter is a batch estimation algorithm that requires a data series with certain number L of data points. As a result, the TV filter is performed on the estimation results from LSPF. A flowchart of LSPF integrated with TV filter is illustrated in Fig. 25.

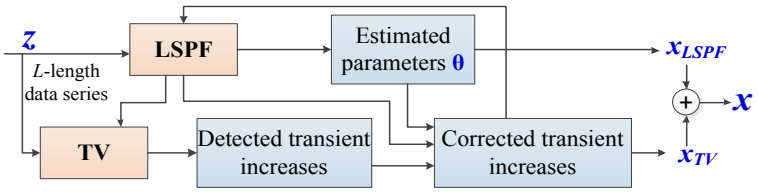


Figure 25. LSPF+TV for degradation tracking and abrupt fault detection

Besides the improvements on PF for tracking and predicting time-varying system performance degradation, another improvement made in this project is to generalize the prognostic modeling with respect to the application scenarios (e.g. different operating and environmental conditions) [3]. The contribution is the generation of generic health parameters that represent the health status of a system/machine and are independent of system/machine operating and environmental conditions. Different from physical health parameters, a dimensionless health parameter within the range of [0 1] has been investigated in this project. A value of 1 is assigned to the initial stage of the new machine/system, whereas a value of 0 is assigned to the end of the machine life. One advantage of this type of health parameter lies in the simple setting of the threshold to determine machine failure. To construct the measurement model [5,6,8,9], mapping that relates the measurements to the health parameter is established by data-driven methods, such as termed extreme learning machine (ELM), support vector machine (SVM), and artificial neural network (ANN).

4.5 Use Cases

We developed four use cases to demonstrate the techniques presented in Sections 4.1 to 4.4.

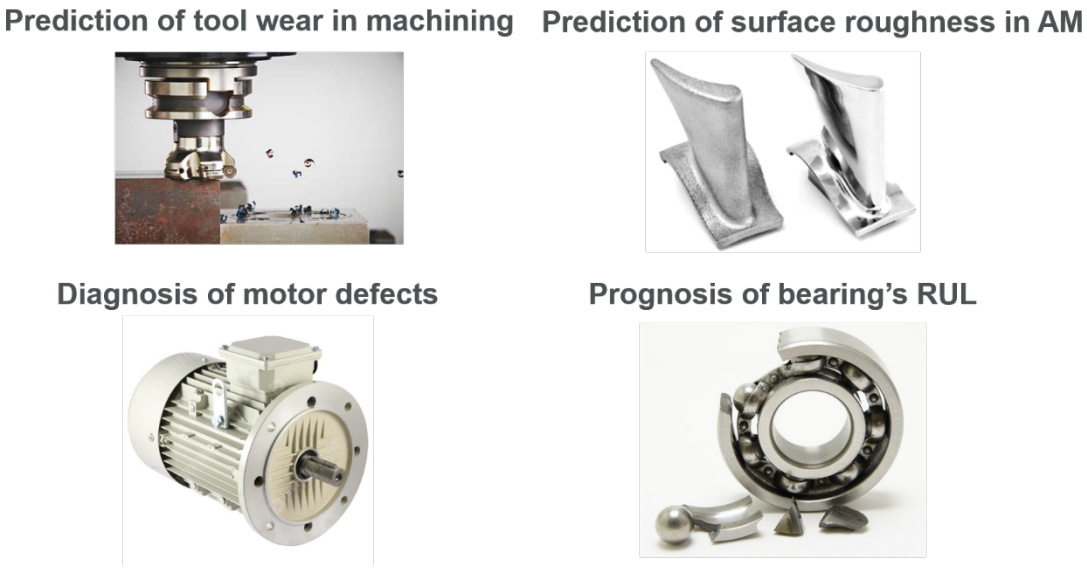


Figure 26. Four use cases: (1) monitoring and prediction of tool wear in milling, (2) prediction of surface roughness in additive manufacturing, (3) diagnosis of motor defects, and (4) prediction of bearing RUL

4.5.1 Use case 1: Monitoring and prediction of tool wear in milling

Experimental Setup

The experiment was conducted on a Röders Tech RFM 760 3-axis high-speed vertical CNC machine. Seven signal channels, including cutting force, vibration, and acoustic emission, were monitored using a Kistler piezoelectric dynamometer, three Kistler piezoelectric accelerometers, and a Kistler acoustic emission (AE) sensor. The piezoelectric dynamometer was mounted on the table of the CNC to collect cutting force data in X, Y, Z dimensions. The piezoelectric accelerometers were mounted on a workpiece to collect vibration data in X, Y, Z dimensions. The AE sensor was also mounted on the workpiece to collect AE data during the milling experiment. AE occurs when a material undergoes irreversible changes (e.g., crack formation or plastic deformation) in its internal structure. Table 1 summarizes the signal channels and measurement data.

Table 1. Signal Channels and Measurement Data

Signal Channel	Measurement Data
Channel 1	F_X: Cutting force (N) in the X axis
Channel 2	F_Y: Cutting force (N) in the Y axis
Channel 3	F_Z: Cutting force (N) in the Z axis
Channel 4	V_X: Vibration (g) in the X axis
Channel 5	V_Y: Vibration (g) in the Y axis
Channel 6	V_Z: Vibration (g) in the Z axis
Channel 7	AE: Acoustic emission (V)

The example cutting force, vibration, and AE signals collected from the dynamometer, accelerometer, and AE sensors are shown in Figures 6, 7, and 8, respectively. Figures 6, 7, and 8 show 127399 sampling signals collected from one cutting test.

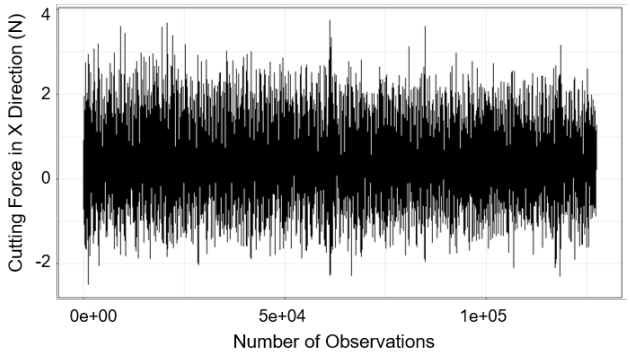


Figure 27. Cutting force in X direction

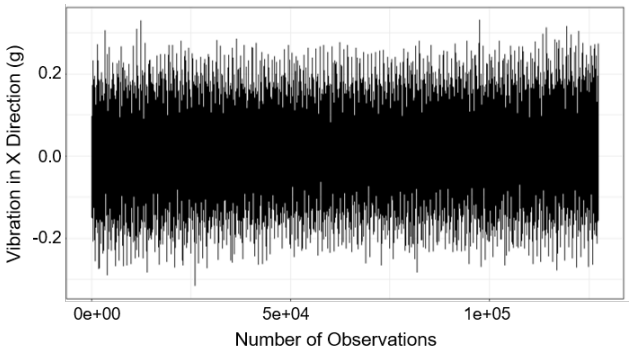


Figure 28. Vibration in X direction

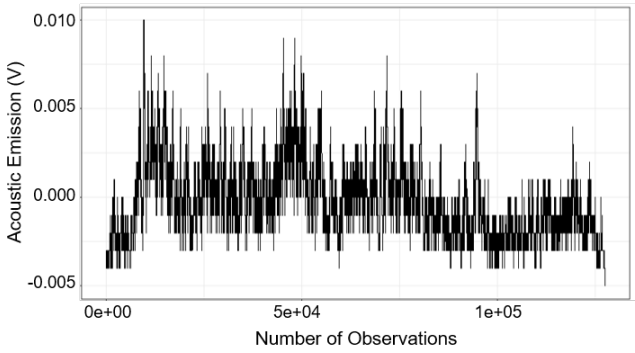


Figure 29. Acoustic emission

The material of the workpiece used in the milling experiment was stainless steel. 315 cutting tests were conducted by the following two steps:

- Remove material from the workpiece using a predefined tool path;
- Measure the amount of tool wear using a LEICA MZ12 high-performance stereomicroscope.

Table 2 summarizes the operating conditions of the milling experiment. The total size of the condition monitoring data collected from 315 cutting tests is 9 GB.

Table 2. Operating conditions of the milling tests

Parameter	Value

Spindle Speed	10400 RPM
Feed Rate	1555 mm/min
Y Depth of Cut	0.125 mm
Z Depth of Cut	0.2 mm
Sampling Rate	50 KHz/channel
Material	Stainless steel

Results and Discussions

Feature Generation and Extraction

In this section, feature generation and extraction are presented. Feature generation involves the process of defining statistical features or variables based on raw data collected from sensors. In this study, a set of statistical features (28 features), including maximum, median, mean, and standard deviation, was generated from the cutting force, vibration, and acoustic emission raw data. The importance of these features for predicting tool wear was evaluated using the variable importance metric expressed in Equation 3.6. Figure 9 shows the variable importance scores for the 14 most important features. The statistical features with greater variable importance scores are more significant. For example, the standard deviation of vibration in the X direction (vb_x_std) with a feature importance score of 231519.6 is the most significant feature.

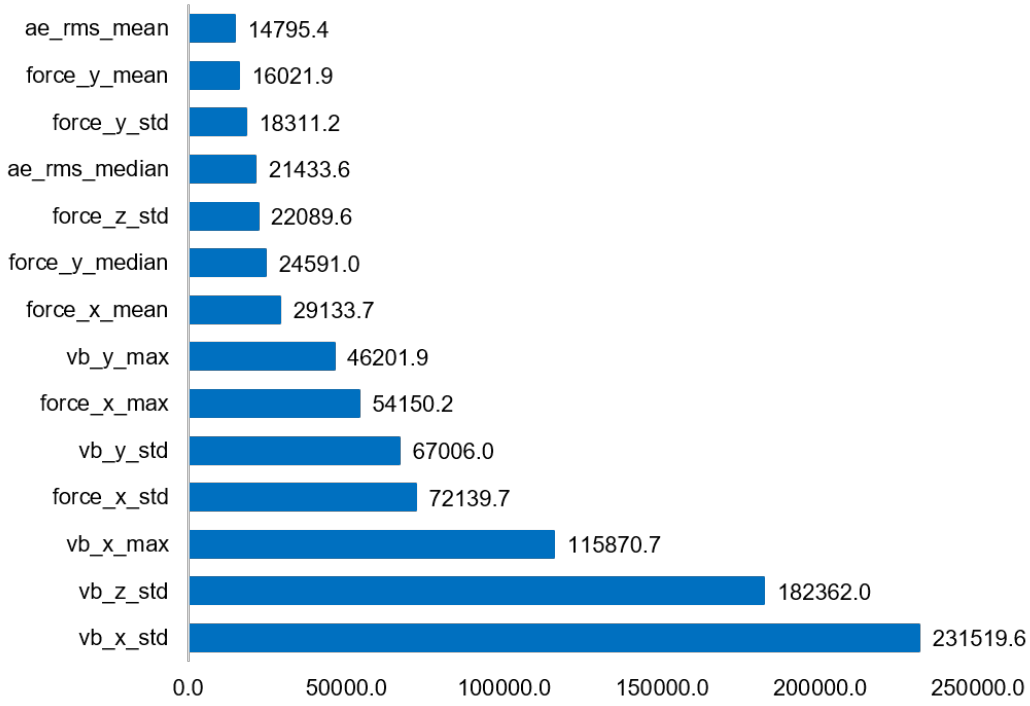


Figure 30. Mean decrease in residual sum of squares/variable importance

Prediction of Tool Wear Based on Random Forests

After generating the statistical features, these feature data are fed into the RFs algorithm. A predictive model for tool wear prediction was trained using 10,000 regression trees. A total of 315 instances in the input data set was divided into training and validation data sets, respectively. To train the predictive model, two thirds of the 315 instances were used for the development of the predictive model. The remainder of the 315 instances was used for model validation. The tool wear prediction results are shown in Figure 10. The data

points in Figure 10 represent observed (true) and predicted tool wear. If all of the data points fall on the straight line in red with a slope of 1, the accuracy of the predictive model is 100%. Figure 10 suggests that the predictive model trained by RFs can estimate tool wear with reasonably good prediction accuracy.

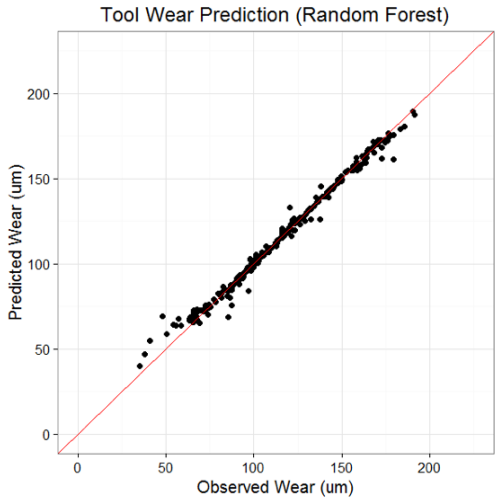


Figure 31. Comparison of observed and predicted tool wear

To measure the performance of the predictive model trained by RFs, several common performance metrics, including mean squared error (MSE), coefficient of determination (R-squared), and training time, were used in this study. The MSE is defined as $MSE = \frac{1}{n}\sum_{i=1}^{n}(\widehat{Y}_i - Y_i)^2$ where $\widehat{Y}_i$ is a predicted value, Y_i is an observed value, and n is the sample size. The MSE measures the average of the squares of the errors. The coefficient of determination is defined as $R^2 = 1 - \frac{SSE}{SST}$ where SSE is the sum of the squares of residuals, SST is the total sum of squares. The coefficient of determination is interpreted as the proportion of the variance in the dependent variable that can be predicted from the independent variable. If the R-squared value is equal to 1, all of the data points fall perfectly on the fitted regression line. If the R-squared value is equal to 0, the model explains none of the variability of the response data around its mean. The R-squared metric provides an indication of the goodness of fit of a set of predictions to the actual values. Table 3 summarizes the MSE, R-squared values, and training time when randomly sampling 50% to 90% of the total data as training data.

Table 3. MSE and R-squared values on test data and training time

	Random forests (10,000 Trees)		
Training size (%)	MSE	R^2	Training time (Second)
50	14.242	0.986	20.876
60	11.466	0.989	26.562
70	10.469	0.990	33.230
80	8.195	0.992	38.995
90	8.295	0.992	45.224

Performance Evaluation for Cloud-Based Parallel Random Forests

The MapReduce-based parallel RFs algorithm was implemented on the Amazon Elastic Compute Cloud (Amazon EC2). Amazon EC2 is a web service that provides scalable high performance computing capacity on the Amazon Web Services (AWS) cloud. In comparison with traditional clusters or supercomputers,

Amazon EC2 runs instances on its physical infrastructure using the open-source virtualization middleware Xen. Various configurations of CPU cores, memory, storage volumes, and operating systems, also known as instance types, are provided on the Amazon EC2 cloud platform. In this study, two instance types were selected to evaluate the performance of the MapReduce-based parallel RFs. Table 4 summarizes the detailed hardware configurations of the C3.8 and R3.8 instances. The C3.8×large instance type has an Intel Xeon E5-2680V2 processor, 32 virtual cores, 60 GB of memory, and 640 GB of solid state drive (SSD) storage. C3 instances are optimized for compute-intensive applications. The R3.8×large instance type has an Intel Xeon E5-2670V2 processor, 32 virtual cores, 244 GB of memory, and 640 GB of solid state drive (SSD) storage. R3 instances are optimized for memory-intensive applications.

Table 4. Hardware configurations for amazon EC2 instances

Instance Type	C3.8×large	R3.8×large
Operating System	Linux	Linux
Processor	Intel Xeon E5-2680 v2 (2.80GHz)	Intel Xeon E5-2670 v2 (2.50GHz)
Number of Virtual CPU	32	32
Memory (GB)	60	244
Storage (GB)	640	640

To evaluate the performance of the MapReduce-based parallel RFs, two performance metrics, including training time and relative speedup ratio, were used. The time to train a predictive model varies depending on the amount of training data and computing capacity. Figure 12 shows the average training time with different amount of training data and cores. For example, the curve in red represents the average training time to train the predictive model with 50% of the total amount of data. Similarly, the curve in pink shows the average training time to train the predictive model with 90% of the total amount of data. The training times are 21, 28, 34, 40, and 47 seconds using one core, respectively. As expected, the training time increases as the training data increase.

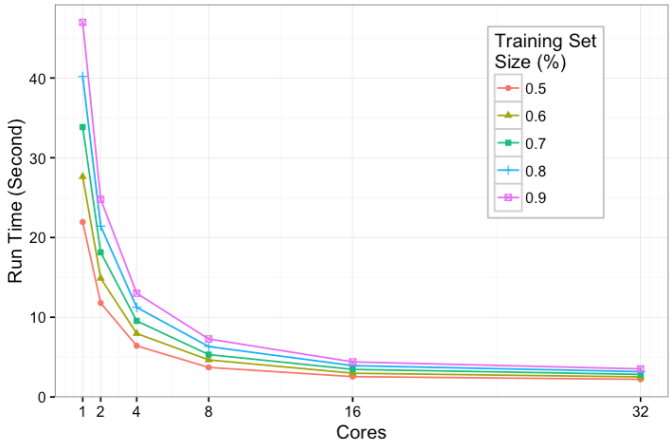


Figure 32. Training time for C3 instances

In addition, to assess the performance of the MapReduce-based PRFs, the predictive model was trained with 1, 2, 4, 8, 16, and 32 cores on the different amount of training data. For example, it took 21, 11, 7, 4, 3, and 2 seconds to train the predictive model with 1, 2, 4, 8, 16, and 32 cores, respectively, when 50% of the total amount of data was used for training. It took 47, 25, 13, 8, 5, and 3 seconds to train the predictive model with 1, 2, 4, 8, 16, and 32 cores, respectively, when 90% of the total amount of data was used for training. Relative speedup ratio measures the relationship between the sequential execution time and the parallel execution time solving the same problem. Figure 13 shows the relative speedup ratios when 90%

of the total amount of data was used as training data. The results show that the MapReduce-based PRFs achieved a near linear speedup for 1 to 8 cores and a sublinear speedup for 16 to 32 cores, respectively.

Because the C3 instance is optimized for compute-intensive applications, the MapReduce-based RPF was also executed on the R3 instance which is optimized for memory-intensive applications. Figure 14 shows the training time. For example, it took 21, 11, 7, 4, 3, and 2 seconds to train the predictive model with 1, 2, 4, 8, 16, and 32 cores, respectively, when 50% of the total amount of data was used for training. It took 44, 23, 12, 7, 4, and 3 seconds to train the predictive model with 1, 2, 4, 8, 16, and 32 cores, respectively, when 90% of the total amount of data was used for training. Figure 15 shows the relative speedup ratios when 90% of the total amount of data was used as training data. The results show that the MapReduce-based PRFs achieved a near linear speedup for 1 to 8 cores and a sublinear speedup for 16 to 32 cores, respectively. The results show that the training time with the R3 instance is almost the same as that of the C3 instance.

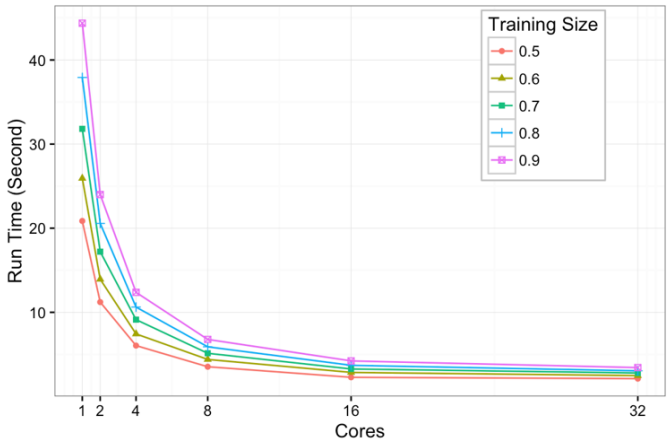


Figure 33. Training time for R3 instances

Summary

In this paper, prediction of flank tool wear in high-speed machining was conducted with RFs and MapReduce-based PRFs algorithms. The MapReduce-based PRFs algorithm was implemented on the Amazon EC2 cloud. The condition monitoring data, including cutting force, vibration, and acoustic emission, collected from 315 milling tests were used to evaluate performance of the algorithms. A set of statistical features was generated as the input of the machine learning algorithms. The performance metrics include *MSE*, R-squared, and training time. The experimental results have shown that RFs can predict tool wear very accurately with the condition monitoring data. The importance of the statistical features can be measured using RFs. In addition, the prediction intervals associated with tool wear predictions were computed to measure uncertainty in tool wear prediction. Moreover, the MapReduce-based PRFs algorithm was developed to increase the efficiency of the original RFs algorithm. The experimental results have shown that a significant increase in training time (15 times with 32 cores) has been achieved by parallelizing the original RFs with two Amazon EC2 instances. In the future, the MapReduce PRFs is implemented on a cloud with multiple computing nodes to evaluate the scalability of the algorithm. Efforts are also focused on evaluating the performance of the algorithm on large volumes of streaming data from multiple CNC machines.

4.5.2 Use case 2: Monitoring and prediction of surface roughness in additive manufacturing

The objective of the second use case is integrate a public HPC cloud with a private cloud for monitoring and predicting the surface roughness of additively manufactured parts with multiple sensors and machine learning.

Experimental Setup

This section presents the experimental setup and design of experiments for collecting real time sensor data in private cloud. As shown in Fig. 2, a commercial desktop 3D printer (MakerBot Replicator Plus) was used as the testbed for this study. Some of the features of the printer include a LCD display, an on-board camera, Wi-Fi, and Ethernet connectivity. The build material is Polyactic Acid (PLA). To monitor the FDM process, five sensors were installed on the printer. Two thermocouples (5TC-GG-K-20-36, Omega) measure the temperature of the table and extruder, respectively. Two accelerometers (ADXL335, Analog Devices) measure the vibration of the table and extruder. An infrared (IR) non-contact temperature sensor (MLX90614ESF-DCI-000-SP, Melexis Technologies) measures the temperature of the deposited material.

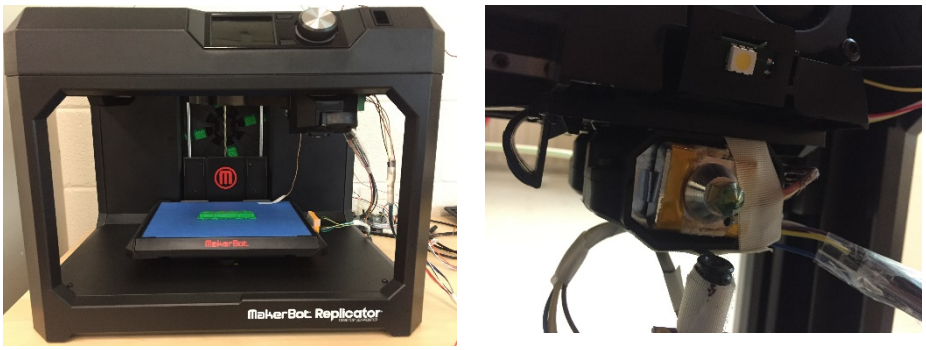
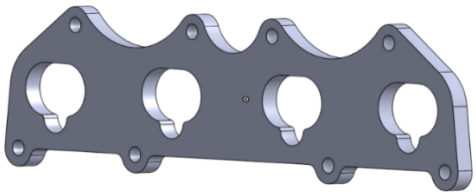


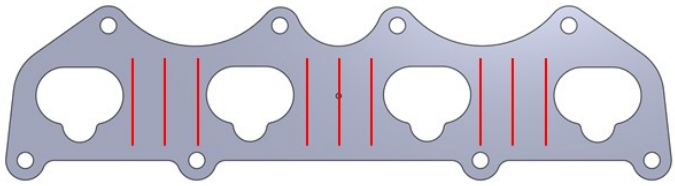
Figure 34. Experimental setup

A contact profilometer is used to measure surface roughness. A profilometer measures small surface variations in vertical stylus displacement as a function of position. A few roughness parameters, including maximum profile peak height, average maximum profile peak height, and maximum roughness depth, can be used to quantify roughness. In this study, roughness average (Ra) was used to quantify roughness. Ra is the arithmetical average of the absolute values of the profile heights over the evaluation length. As shown in Fig. 3 (a), a test part, an engine intake flange, serves as the case study. Fig. 3 (b) shows the engine intake flange printed by the FDM process. Fig. 3 (c) shows how the surface roughness of the 3D printed test part was measured. As shown in Fig. 4, the engine intake flange consists of support, bottom, middle, and top structures with different cross sections.



(a) Engine intake flange

(b) Engine intake flange printed by FDM



(c) Measurement of surface roughness

Figure 35. (a) Engine intake flange; (b) Engine intake flange printed by FDM; (c) Measurement of surface roughness

To validate the proposed approach, a set of experiments was designed. As shown in Table 1, three factors, including layer thickness, extruder temperature, and ratio of print speed to extrusion rate, were selected. To generate training and validation data, a full factorial design of experiments was conducted. Twenty-seven (27) tests were designed. Each test was replicated three times. Eighty-one (81) tests were conducted. Nine (9) channels of sensor data were collected during each test. Five (5) statistical features, including maximum, median, mean, minimum, and standard deviation, in the time domain were extracted from each signal channel. After each test, surface roughness was measured using the contact profilometer.

Table 5. Design of experiments

Factor	**Level 1**	**Level 2**	**Level 3**
Layer Thickness (Mm)	0.20	0.25	0.30
Extruder Temperature (°C)	210	220	230
Print Speed/Extrusion Rate	0.85	1.00	1.15

Results and Discussions

After the sensor data are collected, the statistical features extracted from the entire condition monitoring data in private cloud, and were fed into the machine learning algorithms as input. Features are transferred to public cloud, where multiple machine learning algorithms are trained. After multiple algorithms are trained in public cloud, real time sensor data and features are fed into public cloud continuously to predict surface roughness of additively manufactured parts. To evaluate the performance of the predictive model trained by the algorithms, a 10-fold cross-validation method was used. Cross-validation is a model validation technique for estimating how accurately the predictive model performed. In 10-fold cross-validation, the original dataset was randomly partitioned into ten equal sized subsets. Of the ten subsets, a single subset was retained as the validation data for testing the model, and the remaining nine subsets are used as training data. The cross-validation process was then repeated ten times (10 folds), with each of the ten subsets used exactly once as the validation data. The results from the ten folds were then averaged to produce a single estimation.

Fig. 5 shows the error rates of the predictive models trained on individual sensor measurements using RFs. As the amount of condition monitoring data increases, the relative error rates of the predictive models trained on individual sensor measurements decrease. In addition, during the first 30% of the build time, the 3D printer transitions from an unstable operating condition to a stable operating condition. Therefore, the relative error rates decrease as the build time increases. During the last 70% of the build time, the relative error rates become almost constant. Moreover, the feature-level data fusion method was used to improve prediction accuracy. Data fusion is the process of integrating multiple data sources to produce accurate predictive models. The expectation was that fused data would be more informative than any individual data source. Data fusion techniques generally fall into three categories: data-level, feature-level, and decision-level fusion. Data-level fusion is a lower level fusion method where multiple sensor data sources are fed into a machine learning algorithm directly. Feature-level fusion is an intermediate level fusion method that

requires the integration of extracted features. Decision-level fusion is a high level fusion method that aggregates sensor information after a response variable has been estimated by each sensor. In this study, the feature-level fusion method was used because it has several advantages. First, feature-level fusion is more computationally efficient because it processes extracted features that are more informative instead of raw signals. Second, while decision-level fusion is a higher level fusion method, decision-level fusion requires complex decision rules. As shown in Fig. 5, the predictive model trained by RFs is more accurate by integrating multiple sensor sources. The relative error rate of the predictive model trained using the feature-level data fusion method ranges between 0.082 and 0.044, whereas the relative error rates of the predictive models trained on individual sensor sources range between 0.098 and 0.049.

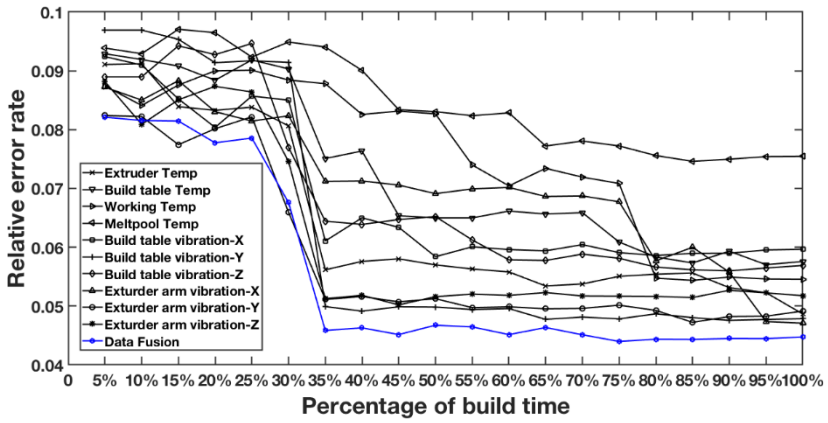


Figure 36. Relative error rates of predictive models trained on individual data source versus multiple data sources

Because the sample part has different cross-sectional structures, we extracted the statistical features from the condition monitoring data collected during the time when different cross-sectional structures were built. For example, we trained five predictive models using RFs and the feature-level data fusion method on the condition monitoring data collected by the time when 20%, 40%, 60%, 80%, and 100% of the support structure was built. As shown in Table 3, the relative error rates of the predictive models are 0.083, 0.082, 0.081, 0.078, and 0.078, respectively. As expected, the accuracy of the predictive models increases as the amount of training data increases. Tables 4, 5, and 6 list the relative error rates of the predictive models trained by SVR, RR, and LASSO using the feature-level data fusion method.

As shown in Fig. 6, the performance of the predictive models trained on varying percentage of build time using RFs, SVR, RR, and LASSO is comparable in terms of the relative error rate. For example, the relative error rates of the predictive models trained on the condition monitoring data collected by the time when 25% of the entire sample part was built range between 0.082 and 0.074. The performance of RFs, SVR, RR, and LASSO increases significantly in the time interval between 25% and 35% of the built time. The relative error rates of the predictive models trained on the condition monitoring data collected by the time when 50% of the entire sample part was built range between 0.047 and 0.042. The prediction accuracy of RFs, SVR, RR, and LASSO becomes almost constant in the time interval between 55% and 100% of built time. The relative error rates of RR and LASSO are slightly less than that of RFs and SVR in the time interval between 55% and 100%.

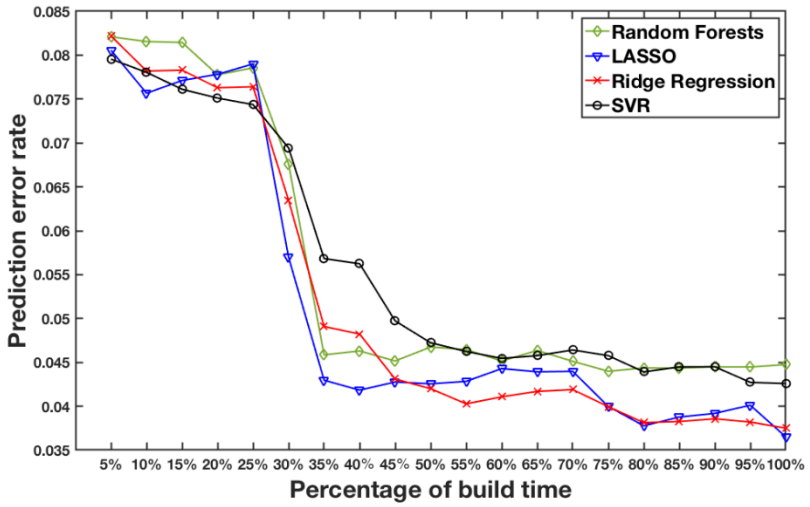


Figure 37. Error rates for RF, SVR, RR, and LASSO using 10-fold cross-validation

Furthermore, LASSO was used to quantify the importance of the statistical features. The regression coefficient (β_j) in Eq. (9) can be used to measure the importance of the statistical features. If a regression coefficient associated with a statistical feature is set equal to zero, then the statistical feature is not used in the data fusion method. If a regression coefficient associated with a statistical feature is greater, then the statistical feature is more important. Fig. 7 shows top ten statistical features. The most important feature is the maximum value of the vibration of the build table in the Z direction, the least important feature is the minimum value of the vibration of the extruder arm in the Y direction.

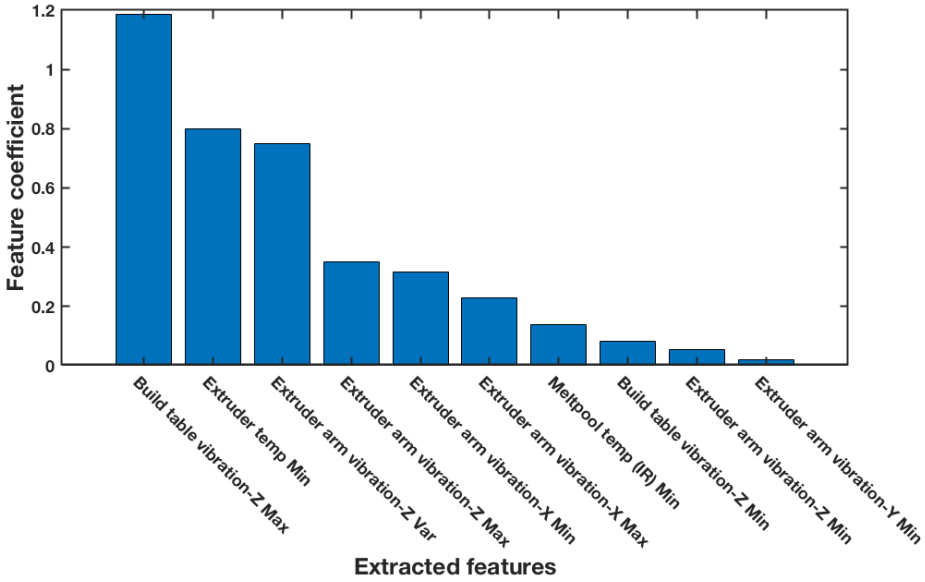


Figure 38. Regression coefficients calculated using LASSO

Summary

We integrated a public HPC cloud with a private cloud and developed a predictive modeling approach to surface roughness prediction in FDM processes using machine learning algorithms. A real-time monitoring system was developed and integrated to a FDM-based 3D printer to monitor the vibration and temperature of the extruder and table as well as the melt pool temperature. A set of statistical features was extracted from the sensor measurements. RFs, SVR, RR, and LASSO were used to train the predictive models on the

individual sensor measurement. In addition, a feature-level data fusion method was used to improve prediction performance by integrating multiple sensor sources. The experimental results have shown that the predictive models trained by the machine learning algorithms on the condition monitoring data predict the surface roughness of additively manufacturing parts with very high accuracy. The performance of these algorithms is comparable in terms of relative error rate.

4.5.3 Use Case 3: Diagnosis of machine structural faults based on sparse representation and dictionary learning

The objective of this section is to demonstrate the effectiveness of the developed diagnosis method using induction motor as a representative manufacturing application. In modern manufacturing, induction motors are widely used in equipment such as belt conveyors, cranes, lifts, compressors, pumps, fans as the main power source. They are one of the most critical parts in manufacturing system, consuming a large portion of total electricity (about 40%) and their failure often leads to immediate shutdown of the production itself. Therefore, effective and efficient diagnosis of induction motor provides the scientific basis for proper maintenance strategy, leading to reduction in unexpected maintenance cost and production downtime, and ultimately contributes to the improved operational efficiency, energy usage and overall sustainability in manufacturing. Due to the complex electro-magnetic and mechanical interactions occurred inside the motor during operation, the sensing signal collected is often noisy, with the relationship to the corresponding health condition being highly non-linear and beyond existing physical knowledge. The developed method based on dictionary learning provides a data-driven solution for effective fault characterization as it adaptively extracts the condition-related patterns from the signal through iterative process, and bypasses the limitation in physical knowledge.

Experimental Setup

The experimental setup is shown in Fig. 39. In total, six motor health status are selected as classification class in this study, as shown in Table 6. The five faulty conditions encompass the most commonly reported induction motor failures in bearing, rotor and stator. The statistics have shown that failure in these three components takes up 85% of all induction motor failures. The belt/pulley system simulates the equipment configuration commonly seen in manufacturing settings. Motors are driven with 50Hz power supply. As the structural fault is expected to be manifested through the change in electrical and vibration signals, due to the electro-magnetic and mechanical interactions, a tri-axial accelerometer is mounted on motor top for vibration sensing, and the electrical current of the motor is measured using a current sensor. The sampling rate is 30kHz [1,7].

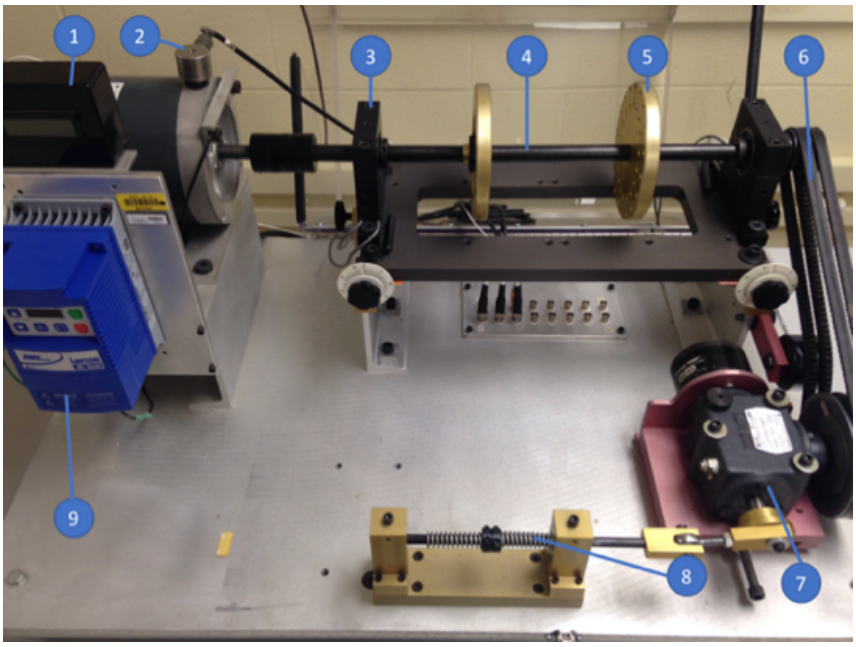


Figure 39. Experiment setup: 1) Tachometer 2) Tri-axial accelerometer 3) Bearing 4) Shaft 5) Load disc 6) Belt 7) Bevel gearbox 8) Reciprocating mechanism 9) Controller

Table 6. Motors used in experiment

Health Status	**Description**
Normal	Healthy condition
Broken rotor bar	3 broken rotor bars
Bowed rotor	Rotor bent in center 0.01”
Unbalanced rotor	Unbalance created by adding 3 washers on rotor
Stator winding defect	3 turns shorted in stator winding
Defective bearing	Inner race defect bearing in shaft end

The historical sensing data collected from each motor are sent to Microsoft Azure cloud platform for the off-line training stage. The scalability of the public cloud provides a means for effective handling of large amount of data from the manufacturing shop floor, often collected from various sensors with high sampling rate. In the cloud platform, these sensing data are first converted to virtual samples, before dictionary learning is applied for fault characterization and classifier construction. Specifically, each individual sensing signal is first split into segments, or samples. The kernel matrix is then computed through the kernel function over the corresponding set of samples. Virtual samples are then obtained by eigen-decomposing the kernel matrix and rearranging its eigen-values and eigen-vectors in the form of symmetrical inner product, as shown in Fig. 21. These virtual samples serve as the input of dictionary learning for the corresponding motor health condition.

As the process of dictionary learning for each motor is independent from the other, parallel computing capability of Microsoft Azure is leveraged to improve the efficiency of the training process. Specifically in this use case, dictionary learning for 6 different motor conditions are performed with each condition utilizing 1 central processing units (CPU). The result is simultaneous learning for all motor health conditions. To demonstrate the computational efficiency improvement enabled by parallel computing, other CPU assignments for dictionary learning are also tested and the total time for dictionary learning from all six conditions is recorded. Once all six dictionaries are obtained, they’re concatenated to construct the

sparse classifier for evaluating the diagnosis performance on the testing signals. Motor condition classification accuracy is used as the performance indicator. The complete flowchart for induction motor fault diagnosis is shown in Fig. 40.

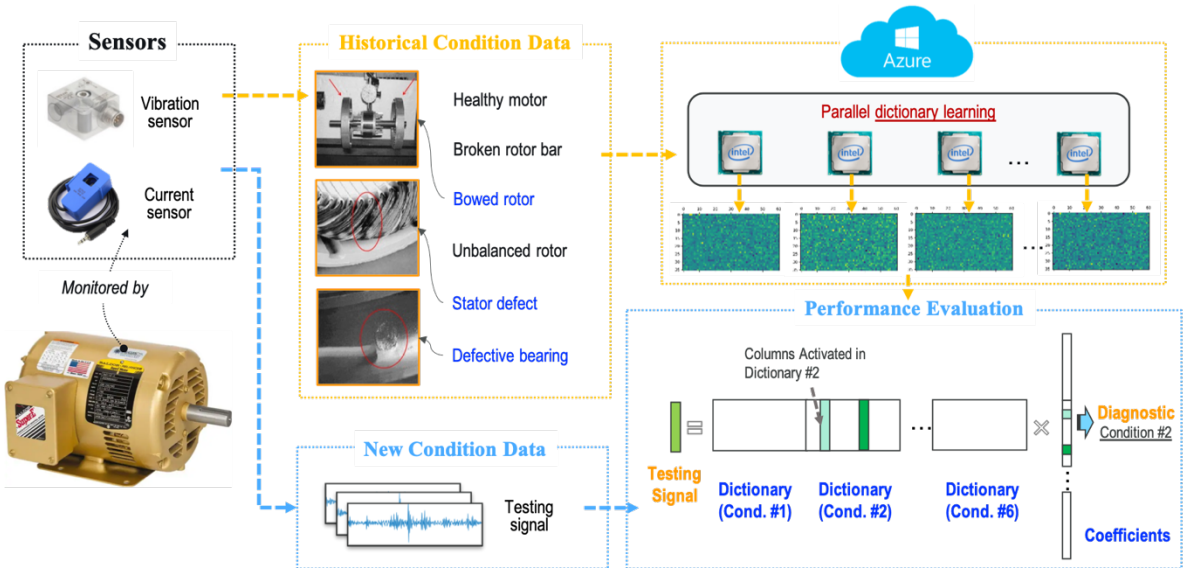


Figure 40. Flowchart: induction motor fault diagnosis

Results and Discussions

The comparison of computational efficiency of parallel dictionary learning using different number of CPU's is illustrated in Fig. 41. As the number of processors increases, the decreasing trend in total dictionary learning time is observed. A 75% reduction (17.4s vs. 4.1s) is achieved when 6 processors are used with each assigned to learn the dictionary of an individual condition, as compared to using just one CPU. It is noted that for the tests in which 3, 4 and 5 processors are used, the computational time stays the same. This is due to the fact that in these three cases, there's always at least one dictionary waiting to be learned in the queue. Therefore, although some CPU's are idol after learning one dictionary, the total computational time in these cases always reflects the processors that learn two dictionaries.

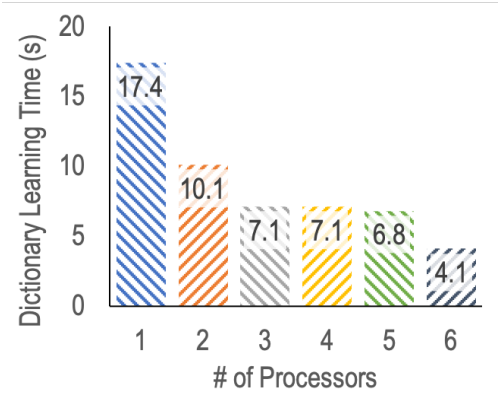


Figure 41. Computational efficiency comparison with difference number of CPU used for dictionary learning

The developed diagnosis method based on dictionary learning and sparse classifier is further compared to other two techniques in terms of induction motor fault classification accuracy, as shown in Fig. 42. The first method is kernel support vector machine (SVM). The second method is based on the deep convolutional neural network (DCNN). It is shown that the developed kernel dictionary learning-based

method outperforms kernel SVM and achieves state-of-the-art classification accuracy while using significantly less training time as compared to the more sophisticated DCNN-based method [7].

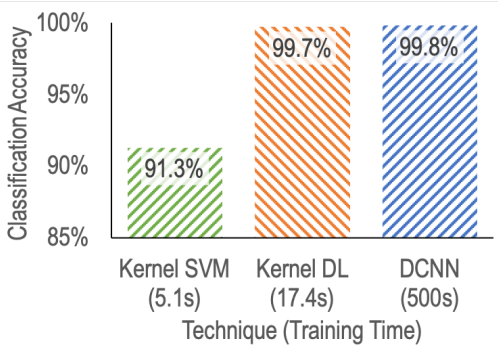


Figure 42. Comparison of induction motor fault classification accuracy and training time using different techniques.

Summary

A machine multi-fault diagnosis method based on dictionary learning and sparse classifier has been developed and evaluated using induction motor as representative manufacturing application. Dictionary learning is able to analyze the sensing signal for condition-related signal patterns extraction. A virtual sample-based kernel method has been developed to improve the capability of the diagnosis method to handle data non-linearity. The efficiency of the developed method has been further improved by leveraging the parallel computing capability enabled by the cloud computing platform. Sparse classifier, constructed using the obtained dictionaries, has been shown to be effective in correctly identifying the structural motor faults in the presented use case.

For the manufacturers, the main benefits of the developed diagnosis method based on dictionary learning and sparse classifier are two-fold: 1) the method can be extended to other manufacturing use cases, as the dictionary learning algorithm, when analyzing the signal for condition-related pattern recognition, does not assume prior knowledge of the manufacturing system and can be adapted to various signals, and 2) the developed method is able to fully utilize the parallel computing capability of the cloud platform, allowing the dictionaries to be efficiently learned and classifier constructed and therefore, suitable for the increasing complexity in modern manufacturing settings.

4.5.4 Use Case 4: Prognosis of machine performance degradation and prediction of remaining useful life (RUL) based on particle filter

The objective of this section is to evaluate the improvement made to particle filter in advancing tracking and prediction of the performance degradation in engineering system, using two datasets. The first dataset consists of aircraft engine performance data obtained from high-fidelity simulation. The engine degradation is often characterized by abrupt performance drop which poses significant challenges for traditional tracking method. This dataset allows demonstration of the capability of the developed integrated method of particle filter and total variation filter in detecting and adjusting the tracking to the abrupt performance drop. The second dataset consists of bearing vibration data obtained through run-to-failure experiment. It is used to demonstrate the capability of multi-mode particle filter in detecting and adjusting to the transition among different degradation stages, such as fault initiation stage and development stage.

Aircraft Engine: Simulated Dataset

To evaluate the performance of the developed PF-based prognostic modeling method, a set of high-fidelity system level engine simulation data has been evaluated. The data set was created with a Matlab Simulink

tool called C-MAPSS, which was designed to simulate normal and faulty engine degradation over a series of flights (cycles). Each flight simulates a representative flight profile. For the normal condition case, the engine is given an exponentially degrading flow capacity and efficiency profile, which denotes the degradation of system performance. The abrupt fault is manifested by increasing the efficiency and flow capacity degradation from the fault time point until the end of the simulation for the remaining flights. After a flight is simulated, a snapshot of all engine parameters is taken in the middle of cruise (engine working under approximately the same operating conditions) and applied to estimating engine state and predicting the degradation trend. Each dataset contains both the simulated measurements and a dimensionless health parameter, which is a function of four parameters: 1) fan stall margin, 2) HPC stall margin, 3) LPC stall margin, and 4) exhausted gas temperature [5,6]. One dataset from each of the normal and faulty cases is utilized as the training dataset to train the ELM network and estimate the relationship between the parameter and measurements. The rest of the datasets are subsequently utilized to test the performance of the developed methods that first compute the dimensionless health parameter from the measurements through the trained ELM network, and then track and predict the propagation of the health parameter through PF. An example of computed health parameter for two normal degradation cases are illustrated in Fig. 43. It is observed that the deteriorations follow an exponential law.

Aircraft Engine: Results and Discussions

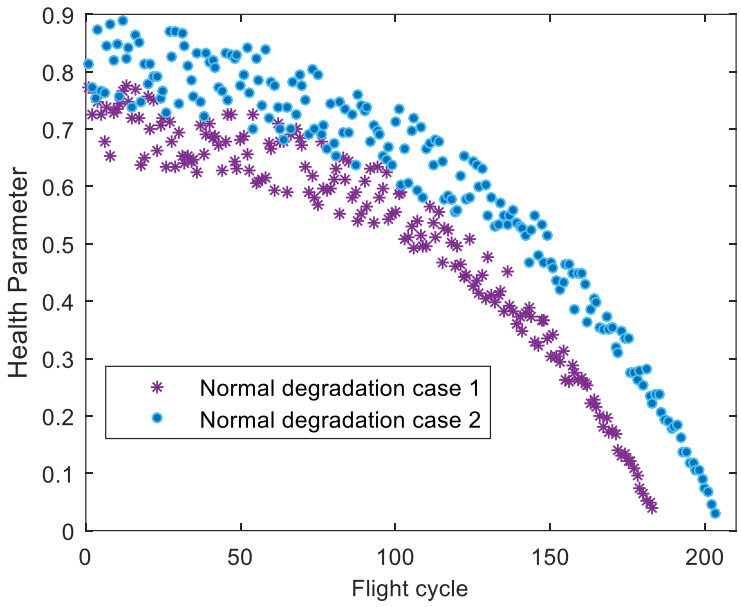


Figure 43. Propagation of dimensionless engine health parameter

Figure 44 demonstrates examples of engine performance degradation prediction for two degradation scenarios: 1) gradual degradation (left figure in Fig. 44) and 2) gradualt degradation and abrupt fault occurences (right figure in Fig. 44). The left figure shows prediction results using measurement data through the 100th flight cycle to tracking and determining the degradation modes. The right figure shows the health parameter estimation and prediction based on joint PF and TV filter for the combined degradation + fault case (i.e. both gradual deterioration and abrupt fault are present). The abrupt fault is introduced at the 27th cycle. The results indicates that the proposed PF+TV filter can reliably track both gradual degradation and abrupt changes. It should be mentioned that the median of the estimation and estimation paths are plotted based on the last update of the parameters during each stage. During the tracking stage, the update based on new measurements reduces the state and parameter estimation uncertainty caused by process noise. However, during the prediction stage, the effect of process noise on predicted health parameter would

accumulate, since no updated information is available from new measurements, leading to the increased prediction uncertainty and confidence limits over time.

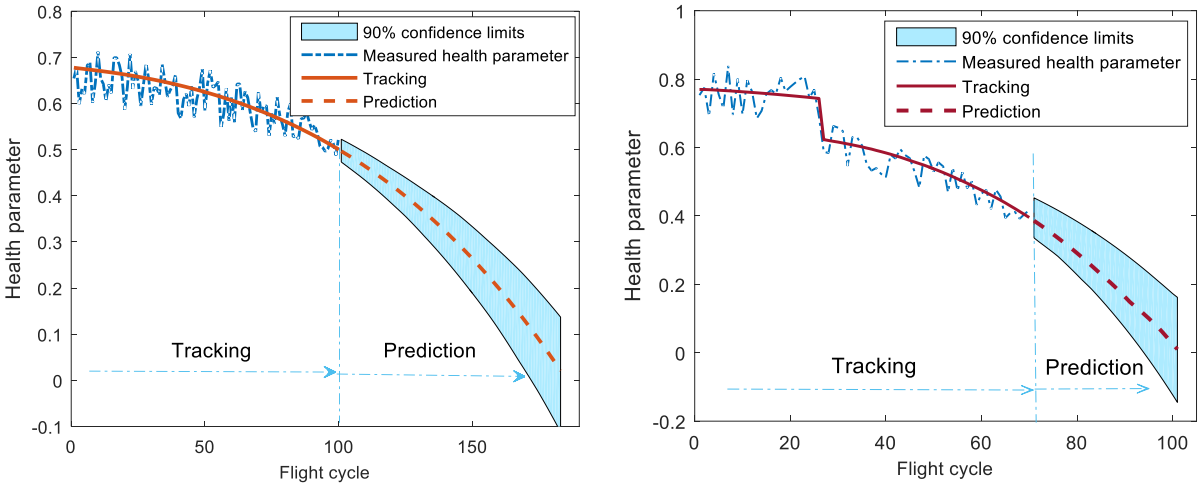


Figure 44. Engine performance degradation tracking by PF+TV

In this project, the performance comparison between the developed PF method and Extended Kalman filter (EKF) has been conducted, as EKF has been widely applied to engine performance tracking, for both the normal and faulty cases. As can be seen in Fig. 45, tracking and prediction by using the PF technique (PF+TV) delivered higher accuracy than that by EKF, in both the normal and faulty cases. Estimation using EKF has shown to deviate significantly from the true path when transient change is present, whereas the PF technique has stayed on track. To illustrate the difference quantitatively, Table 7 compares the estimation error between the PF and EKF technique, through a Monte Carlo simulation. Monte Carlo simulation study is conducted to demonstrate the robustness of the developed method based on PF. Each scenario (normal degradation and fault degradation shown in the Fig. 44) has been run for 500 times. The results are represented by the root mean square error (RMSE) of median predictions.

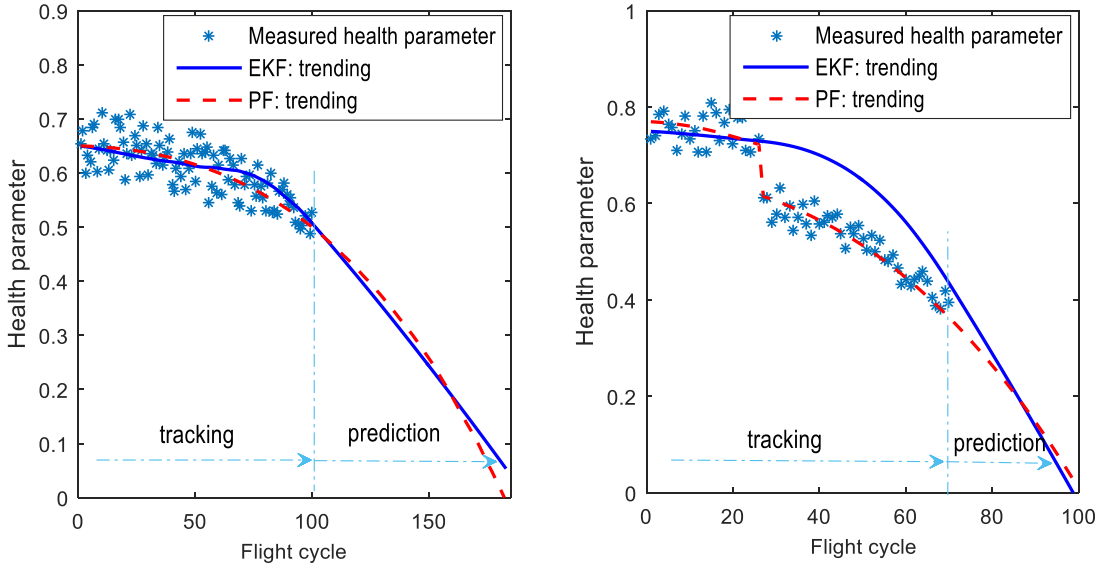


Figure 45. Performance comparison between PF and EKF

Table 7. Performance comparison on estimation error between the PF and EKF technique

	Gradual deterioration	Gradual deterioration and aburpt fault
PF	0.9%	1.0%
EKF	3.4%	8.7%

Bearing: Experiment Setup

To verify the effectiveness of the multi-mode PF, vibration data collected from two run-to-failure bearing experiments have been analyzed. The test system used in the experiments is shown in Fig. 46. The experiments were conducted under constant rotational speed at 2,000 rpm and a radial load of 6,000 lb. Four Rexnord ZA-2115 double row bearings were installed on the shaft and all bearings were force lubricated. The shaft is driven through sheave/belt transmission by an AC motor. A magnetic plug was placed in the oil feedback pipe to collect debris from the oil recycle process. Test stops when the accumulated debris adhered to the magnetic plug exceeds a predefined level and causes a switch to turn off. A PCB 353B33 high-sensitivity quartz Integrated Circuits Piezoelectric (ICP) accelerometer was installed on each bearing housing to measure the vibration. The vibration data were collected every 10 minutes, with 20 kHz sampling rate, and used as indicator of bearing health condition for subsequent performance tracking and remaining life prediction [2,10].

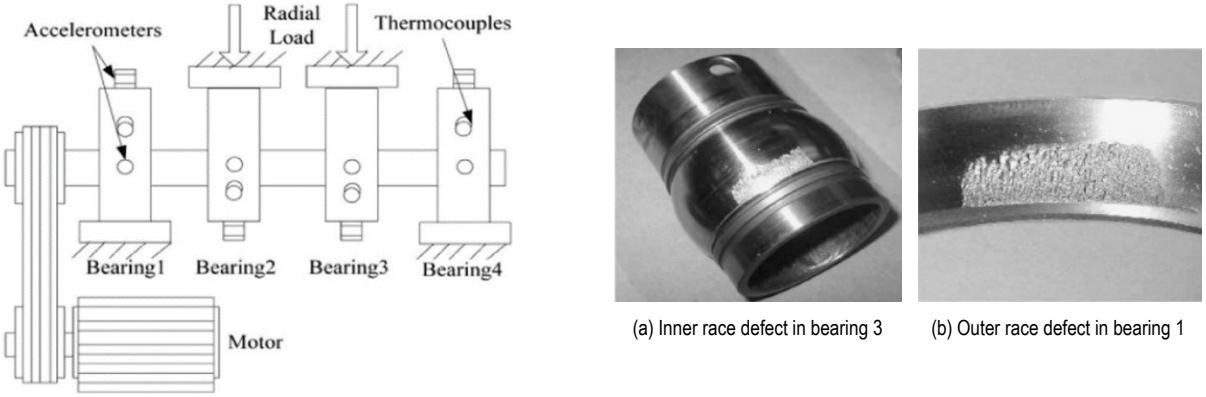


Figure 46. Machine setting and illustration of bearing defects

Bearing: Results and Discussions

Figure 47 illustrates the tracking results of vibration variation due to bearing performance degradation by the two-mode PF and an extended Kalman filter (EKF), for the inner-race and outer-race defect, repectively. For the inner-race defect (left figure), as the vibration before 21,000 minutes is maintained at a similar level, figure 47 screenshots the tracking result after 19,800 min. The performance tracking by the two-mode PF is implemented by 2000 particles. It is noted from the figure that the bearing performance stage 2 (*i.e.* defect initiation stage) starts from 21000 minutes, and the stage 3 (*i.e.* accelerated defect growth) starts after 21330 minutes. For the outer-race defect (right figure), the stage 2 starts from 7040 minutes, and stage 3 shows an exponential crack growth after 9450 minutes. The tracking by the two-mode PF is demonstrated to be robust to the vibration variation caused by the bearing performance degradation and stage transition, whereas the EKF is not sensitive to the abrupt performance variation and thereby lose tracking in stages 2 and 3. The particle distributions provide quantification of uncertainties associated with the performance degradation, using a measure of confidence interval. In this paper, the failure threshold for vibration RMS is defined as 0.4.

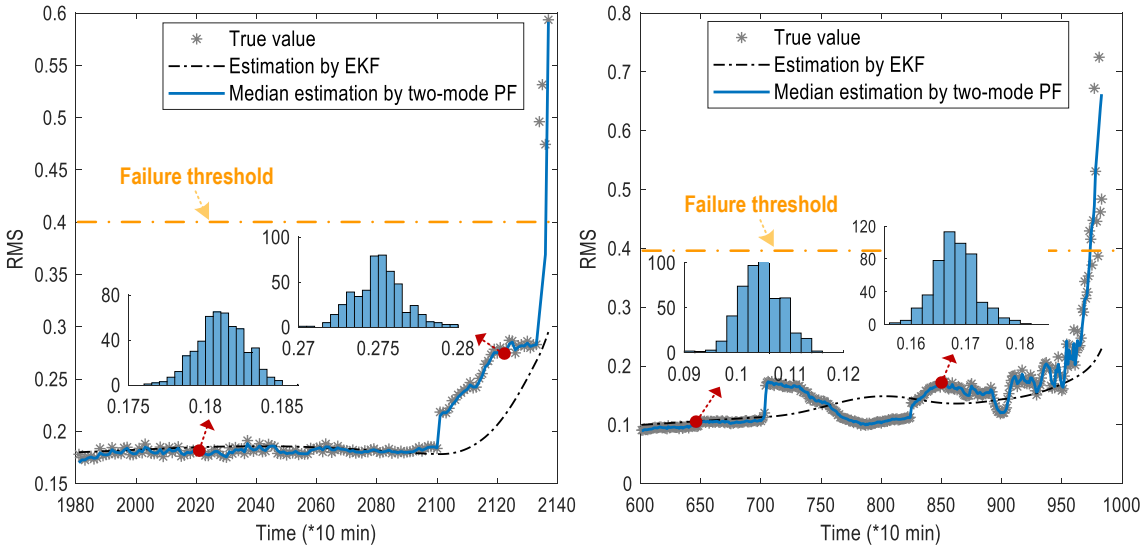


Figure 47. Performance tracking by PF

Figure 48 illustrates the mode transition during the tracking process by the two-mode PF. The onset of stage 2 is indicated by the first transition from Mode 1 to Mode 2, which means that there is a jump in the measured vibration data. Except for the start time of stage 2, the rest data points in stage 2 can still be characterized by Mode 1. This means that the variation of vibration caused by the defect initiation is relatively slow, compared to the vibration variation caused by accelerated spall propagation, in which the fracture of the bearing inner surface and removal of small, discrete particles of material worsen in an accelerated speed. The onset of stage 3 is indicated by the time when successive Mode 2 is turned on for system tracking [2,10].

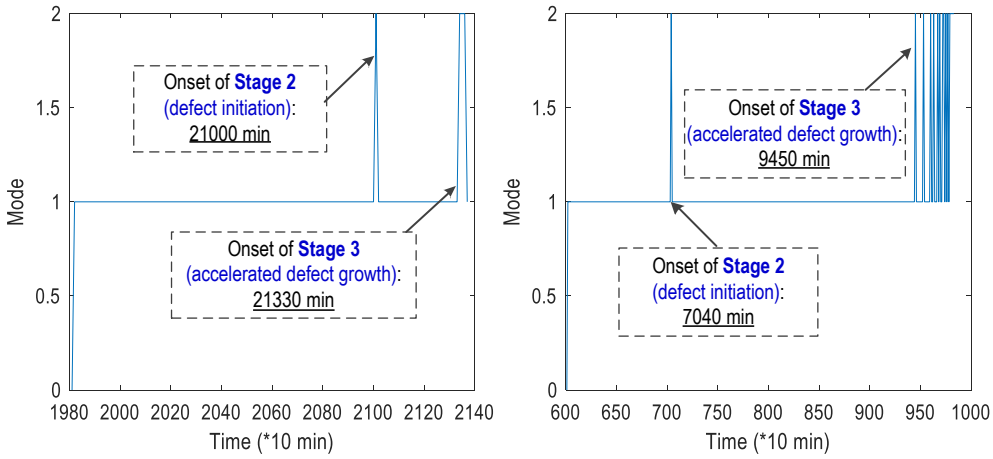


Figure 48. Evolution of mode transiton in the two-mode PF

The RUL prediction by the two-mode PF is shown in Fig. 49. For example, in the inner-race defect case (left figure), the RUL obtained at 21,000 min is predicated upon the updated linear degradation model. It should be noted that the RUL prediction is made only after the onset of stage 2, namely after 21,000 min. 90% confidence bounds of the predicted RUL are provided. The true RULs fall in the prediction bounds, which proves the effectiveness of the developed two-mode PF in predicting the bearing remaining service lives.

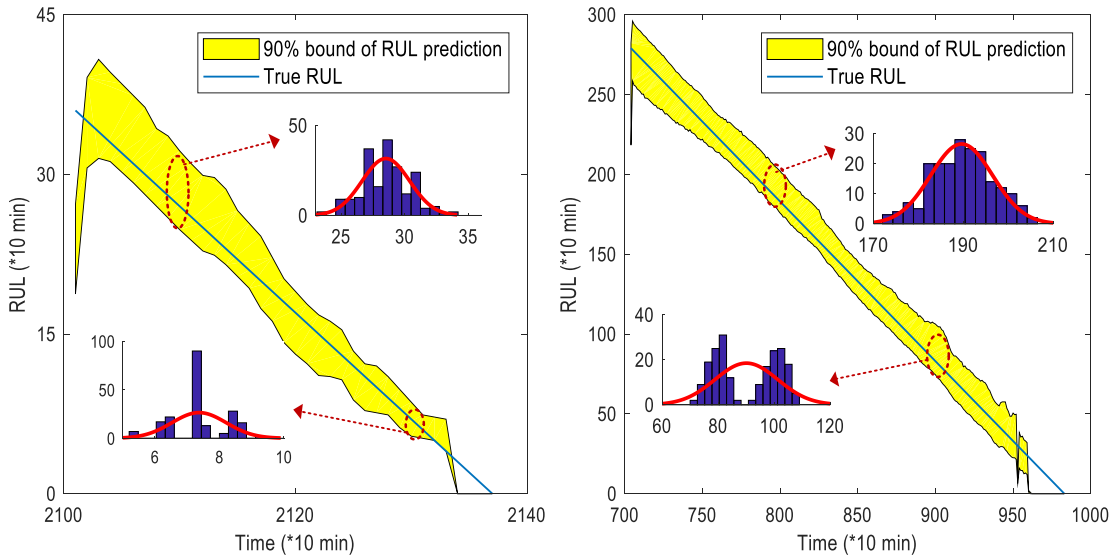


Figure 49. Bearing RUL prediction by the two-mode PF

A performance comparison among EKF, standard PF, and the multi-mode PF is shown in Table 8 [2,10].

Table 8. Performance comparison among EKF, standard PF, and the two-mode PF

	Test 1	Test 2
EKF	12.01	10.77
Standard PF	1.32	1.52
Multi-mode PF	0.41	0.61

Summary

Four improvement has been made to the particle filter in this project: 1) a local search particle filter (LSPF) with adaptive resampling strategy, to address the particle degeneracy problem, for improving the estimation accuracy and narrowing down the confidence interval of estimation and prediction; 2) a multi-mode switching particle filter, to perform time-varying degradation tracking, with the switching between modes automatically realized with the Bayesian framework; 3) an integrated particle filter with total variation filter, to track the gradual deterioration and at the same time detect abrupt performance changes and 4) construction of generic health parameter to generalize the prognostic method with respect to the application use cases. In this section, the impact of these improvement in advancing performance tracking and remaining useful life prediction has been evaluated, using the data from aircraft engine and bearing, and good performance has been demonstrated as compared to the standard particle filter and Kalman filter.

For the manufacturers, the main benefits of the advancement made to the prognosis method based on particle filter are two-fold: 1) the improved system performance tracking capability, especially in the case of multi-mode degradation and abrupt performance change, allows a more accurate estimation of the future evolvement of the system performance and prediction of remaining useful life, serving as the foundation for predictive maintenance and 2) the capability of use case generalization allows the manufacturer to effectively extend the method to a broad range of manufacturing applications, without having to develop specialized, ad-hoc prognosis methods.

Publications (CWRU)

[1] Zhang, J., Wang, P., Sun, C., Yan, R., & Gao, R. X. (2017, October). Induction Motor Fault Diagnosis and Classification Through Sparse Representation. *ASME 2017 Dynamic Systems and Control Conference* (pp. V002T04A005-V002T04A005). doi:10.1115/DSCC2017-5259.

[2] Wang, P., Yan, R., & Gao, R. X. (2018, June). Multi-Mode Particle Filter for Bearing Remaining Life Prediction. *ASME 2018 13th International Manufacturing Science and Engineering Conference* (pp. V003T02A031-V003T02A031). doi: 10.1115/MSEC2018-6638.

[3] Wang, P. & Gao, R. X. (2018, Oct). Lévy Process-Based Stochastic Modeling for Machine Performance Degradation Prognosis. *The 44th Annual Conference of the IEEE Industrial Electronic Society*. doi: 10.1109/IECON.2018.8592928.

[4] Zhang, J., Wang, P., Gao, R. X., & Yan, R. (2018). An Image Processing Approach to Machine Fault Diagnosis Based on Visual Words Representation. *Procedia Manufacturing*. **19**: 42-49. doi: 10.1016/j.promfg.2018.01.007.

[5] Zhang, J., Wang, P., Yan, R., & Gao, R. X. (2018). Long Short-term Memory for Machine Remaining Life Prediction. *Transactions of the SME, Journal of Manufacturing Systems*. **48**: 78-86. doi: 10.1016/j.jmsy.2018.05.011.

[6] Zhang, J., Wang, P., Yan, R., & Gao, R. X. (2018). Deep Learning for Improved System Remaining Life Prediction. *Procedia CIRP*. **72**: 1033-1038. doi: 10.1016/j.procir.2018.03.262.

[7] Zhang, J., Wang, P., Gao, R. X., Sun, C. & Yan,R. (2018). Induction Motor Condition Monitoring for Sustainable Manufacturing. *Procedia Manufacturing*, in press.

[8] Zhang, J., Wang, P., & Gao, R. X. (2018). Modeling of Layer-wise Additive Manufacturing for Part Quality Prediction. *Procedia Manufacturing*. **16**: 155-162. doi: 10.1016/j.promfg.2018.10.165.

[9] Zhang, J., Wang, P., & Gao, R. X. (2019). Deep Learning-based Tensile Strength Prediction in Fused Deposition Modeling. *Computers in Industry*. **107**: 11-21. doi:/10.1016/j.compind.2019.01.011

[10] Wang, P., Yan, R., & Gao, R. X. Multi-mode Particle Filter for Bearing Remaining Life Prediction (submitted, under revision)

5. Accessing the Technology

The developed research method is generic and can be shared within the DMDII community. These methods are pervasive and can be applied to almost all manufacturing machines and processes. It had been designed to be system agnostic even though specific tools are used within the program for the purpose of demonstration of the concept.

In Task A, we developed (1) an interoperable data acquisition system and (2) a scalable computing platform for collecting and preprocessing large volumes of condition monitoring data. The unique characteristics of the interoperable data acquisition system are as follows:

- Open and accessible – leverage open-source SW and COTS as much as possible

- Low Cost – low development cost and total cost of ownership
- Plug-and-Play – enable "drop-in" for data collection, analytics, and more
- Fault tolerant – high availability and high assurance
- Extensible – ease of use for "app" development
- Scalable – can easily scale up and down for resource management
- Ease of setup – easy to setup and maintain

In Task B, we integrated Microsoft Azure cloud with the interoperable data acquisition system and the private cloud storage for transforming big data into intelligent decisions with big data analytics. The algorithms developed in Task B were demonstrated in two use cases presented in Sections 4.5.1 and 4.5.2. The unique characteristics of the cloud-based data processing system are as follows:

- High performance cloud computing power
- Real-time data collection
- Real-time predictive analytics
- Cloud-based parallel machine learning for large volumes of condition monitoring data

In Task C, we developed a dictionary learning algorithm for sparse diagnosis and nonlinear sparse multi-faults classifier. The dictionary learning algorithm was demonstrated in a use case presented in Section 4.5.3. The unique characteristics of these algorithms are as follows:

- The proposed method can find a sparse representation of the input data in the form of a linear combination of atoms
- The kernel method maps the signals from low-dimensional input space into high dimensional feature space for handling non-linearity

In Task D, we developed (1) an advanced degradation tracking method under transient changes and (2) a regularized prognosis model for tracking system performance degradation and predicting the remaining useful life. The method introduced in Task D was demonstrated in a use case presented in Section 4.5.4. The unique characteristics of these algorithms are as follows:

- Track performance degradation under varying operating conditions using the PF+TV filtering method
- Predict the RUL of bearings with better performance

DMDII members can implement the techniques we developed into their manufacturing systems with similar machines or manufacturing process. There is no specific system requirement because it completely depends on the scope and the detail level involved in the application.

Innovation: The key technological advance that has been made during this project is that we addressed one of the primary challenges in the field of manufacturing which is legacy manufacturing machines and CNC machines lack fault and failure detection, self-diagnosis, and predictive maintenance capabilities. We developed a generic framework for cloud-based online machine and process monitoring, diagnosis, and prognosis. We also developed a private cloud-based data acquisition system that collects massive data from machines and processes using the ICT infrastructure that is solely operated within a corporate firewall. In addition, we developed a hybrid cloud platform that integrates the cloud-based data acquisition system with a public high-performance cloud computing system. Moreover, we developed parallel and distributed machine learning algorithms for online diagnosis and prognosis in additive and subtractive manufacturing as well as motors and bearings.

6. Industry Impact & Potential

The developed framework is to integrate cloud computing, smart sensor networks, and parallel data mining and machine learning into online machine and process monitoring, diagnosis, and prognosis. The specific objectives of this project are to 1) develop a generic framework for cloud-based online machine and process monitoring, diagnosis, and prognosis, 2) develop a pilot, private cloud environment for acquiring massive data collected from machines and processes of a corporation, using the cloud infrastructure that is solely operated within a corporate firewall under the control of the corporate ICT department, 3) develop a hybrid cloud prototype that integrates the private cloud with public HPC cloud infrastructure so that the private cloud conducts data collection, screening and cleaning while the public cloud performs computational-intensive data training and visualization, and 4) develop parallel and distributed data mining and machine learning algorithms for online diagnosis and prognosis of representative manufacturing machines and systems. The overall benefit of this developed framework is to perform digital manufacturing more effectively and efficiently in the distributed and collaborative environment.

Any manufacturing systems requiring cloud-enabled machine and process monitoring, diagnosis, and prognosis can take advantage of the resulting framework. It not only can be applied to 3D printing, bearing, and spindle verified in case demonstrations, it can also be used for performing diagnosis and prognosis on many other manufacturing systems.

7. Tech Transition Plan & Commercialization

To help management in making decisions concerning the transition of technology and reduce the technical and cost risks associated with cloud-enabled machines with data-driven intelligence, a process for measuring technology maturity and ensuring that technologies are sufficiently mature before being brought into market is required. In this project, the technology readiness levels (TRLs) defined by Department of Defense was employed in helping to make effective critical decisions. According to the TRL definitions, TRL 4 is referred to as the TRL level on which a system prototype has been validated in a laboratory environment. Before this project, the TRL of cloud-enabled machines with data-driven intelligence is on Level 4 because it has been demonstrated that the private cloud developed at GE and the public cloud developed by Microsoft can collect online real-time data streams and generate big data analytics and data visualization, respectively. After this project, the TRL of cloud-enabled machines with data-driven intelligence is on Level 6 because the prototype system has been tested in a high-fidelity laboratory environment.

With respect to commercialization, the generic framework and prototype are shared with the broad DMDII membership. This project also develops a knowledge base of guidelines and training for future use by academia and industry to implement machine intelligence into legacy and general purpose CNC machines as well as promote the adoption of relevant sensing systems and cloud computing technologies into machines and manufacturing systems.

8. Workforce Development

The educational and outreach objective of this proposal is to broaden the participation of undergraduate, graduate students, and the DMDII consortium members into cloud-based online machine and process monitoring, diagnosis, and prognosis, train future manufacturing engineers to develop a globally competitive workforce, and disseminate research results to the broader communities. Specifically, the targeted audience of our workforce development and education program include undergraduate and graduate students as well as DMDII industry and academia members. We organized a special session on

cloud-based smart manufacturing at the International Manufacturing Science and Engineering conference. This special session provided students, OEMs, SMEs, and large manufacturers with the foundation needed for cloud-based online machine and process monitoring, diagnosis, and prognosis. The special session covered the following topics:

- Cloud-based online machine and process monitoring, diagnosis, and prognosis;
- IoT-enabled real-time data acquisition software and hardware;
- Data-driven predictive modeling in smart manufacturing;
- Data visualization for diagnosis and prognosis;
- Use cases.

Moreover, the investigators disseminated leading-edge research through publications in high-quality peer-reviewed journals such as ASME transactions, SME transactions and present research results at national and international forums and conferences (e.g., SME North American Manufacturing Research Conference and ASME Manufacturing Science and Engineering Conference).

9. Conclusions/Recommendations

The overcharging goal of this research was to integrate cloud computing, low-cost sensors, machine learning, and signal processing techniques into manufacturing equipment for online machine and process monitoring, diagnosis, and prognosis. The following objectives of this project were achieved:

- We developed a generic framework for cloud-based online machine and process monitoring, diagnosis, and prognosis;
- We developed a private cloud-based data acquisition system that collects massive data from machines and processes using the ICT infrastructure that is solely operated within a corporate firewall;
- We developed a hybrid cloud platform that integrates the cloud-based data acquisition system with a public high-performance cloud computing system;
- We developed parallel and distributed machine learning algorithms for online diagnosis and prognosis in additive and subtractive manufacturing as well as motors and bearings.

Specifically, an interoperable sensing system consisting of "drop-in" sensor nodes, a gateway device, and pre-configured "protocol adapters" for plug-and-play fieldbus communications were developed to address machine connectivity and data collection. A container-based private cloud infrastructure that provided a petabyte-scale, high performance, and low latency distributed file system as well as a scalable cloud computing environment with real-time stream analytics, data visualization, and parallel machine learning tools were developed for processing high volume and high-speed data streams. A sparse representation-based classification method was developed and implemented in the hybrid cloud system to diagnose multiple fault sources. A particle filter-based approach was developed to predict the system performance and remaining useful life of manufacturing machines. Four use cases were also developed to demonstrate the cloud-based data acquisition system as well as model-based and data-driven machine learning algorithms. The final project deliverables include:

- An interoperable data acquisition and on-premise cloud computing platform providing scalable data collection and processing for hundreds of manufacturing machines on factory floors;
- A public cloud platform integrated with on-premise private cloud for processing real-time data streams, executing parallel machine learning algorithms, generating big data analytics, and visualizing data;
- A set of experimentally tested algorithms enabling data-driven intelligence for online machine fault diagnosis and prognosis in various types of manufacturing machines and processes, executable on a hybrid cloud computing platform.

10. Lessons Learned

Some of the key lessons learned are described in the following:

- **Computational efficiency consideration for real-time fault diagnosis.** The selection of mathematical approach for computing sparse representation has direct impact on the computational efficiency of the diagnostic method, in particular for real-time applications. Experimental evaluation has shown that the computational time required by the greedy approach for evaluating a testing signal (<0.1s) is significantly less than the LASSO approach (~3 s), making the it suitable for real-time fault diagnosis. It also takes less time than the two other learning techniques evaluated: support vector machines and the neural network-based method, such as DCNN. While the greedy approach is known as an approximation method to the sparse representation problem, the results from the experimental evaluation show that the numerical requirement for *exact* optimization can be relaxed while maintaining the performance, to allow improvement in efficiency for real-time applications.
- **Impact of kernel selection on robustness of diagnosis method.** The selection of kernel function has direct impact on the robustness of the developed diagnostic method. Among commonly reported kernels, the radial basis function (RBF) stands out as a good choice. Evaluations conducted on a wide range of RBF parameter values has shown that the diagnosis accuracy is insensitive to the value selection. The RBF has also shown to be insensitive to the parameter selection as compared to other kernels such as the polynomial kernel. This makes it a generally applicable choice for the data-driven algorithms. These results can serve as guidance for kernel function selection in future research.
- **Use case selection for evaluation of algorithms.** The originally planned project use case is CNC spindle. Due to significant delay in the launch of the project, access to the CNC spindle at GE was lost. As an alternative, three phase induction motors and rolling bearings were chosen for case study, given their close association with the structural dynamics of the spindle system (rotor-shaft assembly and bearing support). The evaluated structural faults (e.g. bowed rotor, unbalanced rotor, bearing inner-race fault etc.) represent the general cases of fault occurrences and degradation in rotary machines. The use case studies have allowed the research team to comprehensively evaluate the capability of the developed algorithms, in spite of the unexpected challenges due to unavailability of industry provided scenarios.
- **Computation time affected by the number of features.** The number of features extracted in private cloud has a significant impact on the computation time of the diagnostic and prognostic method. Empirically, the more features are fed into machine learning algorithms, the more computation time it will cost. More specifically, the computation time increase exponentially when the number of features increase linearly. In use case 2, the computation time is around 1 second using LASSP if only time-domain features are extracted, and the computation time changes to 4 seconds using LASSO if both time-domain and frequency-domain features are extracted. However, the prediction accuracy is also affected by the number of features, the more features the higher prediction accuracy. Therefore, we learn a lesson that a tradeoff needs to be found to balance the computation time and prediction accuracy even using HPC public cloud.
- **Selection of machine learning algorithms for prognostics.** Different machine learning algorithms has different prognostics performance. To improve the performance of diagnosis or prognosis, we should select the machine learning algorithm properly. For example, random forests are used to predict surface roughness in use case 2. However, the computation efficiency should be also considered when select machine learning algorithms. For example, random forests' computation time is around 5 seconds with time-domain features, but other machine learning algorithms takes around 1 second. Therefore, we learn a lesson that machine learning algorithms selection should consider both prediction accuracy and computation efficiency.

11. Definitions and Appendices

What follows are a set of definitions, terms, and acronyms used in this document. These definitions were gathered from various sources including the internet, reference papers, standards organizations, and the authors of this document.

- Cloud computing: Cloud computing is the on-demand availability of computer system resources, especially data storage and computing power, without direct active management by the user.
- Public cloud: A cloud is called a "public cloud" when the services are rendered over a network that is open for public use.
- Private cloud: Private cloud is cloud infrastructure operated solely for a single organization, whether managed internally or by a third party, and hosted either internally or externally.
- Machine diagnosis and prognostics: Machine fault diagnostic and prognostic techniques have been the considerable subjects of condition-based maintenance system in the recent time due to the potential advantages that could be gained from reducing downtime, decreasing maintenance costs, and increasing machine availability.
- Machine learning: Machine learning (ML) is the scientific study of algorithms and statistical models that computer systems use in order to perform a specific task effectively without using explicit instructions, relying on patterns and inference instead.
- Random forests: Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.
- LASSO: LASSO is a regression analysis method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the statistical model it produces.
- Ridge regression: Ridge Regression is a technique for analyzing multiple regression data that suffer from multicollinearity.
- SVR: Support Vector Machine can also be used as a regression method, maintaining all the main features that characterize the algorithm (maximal margin).
- Sparse dictionary learning: Sparse dictionary learning is a representation learning method which aims at finding a sparse representation of the input data (also known as sparse coding) in the form of a linear combination of basic elements as well as those basic elements themselves.
- RUL: Remaining useful life (RUL) is the length of time a machine is likely to operate before it requires repair or replacement.
- Particle filter: Particle filters or Sequential Monte Carlo (SMC) methods are a set of Monte Carlo algorithms used to solve filtering problems arising in signal processing and Bayesian statistical inference.
- EKF: In estimation theory, the extended Kalman filter (EKF) is the nonlinear version of the Kalman filter which linearizes about an estimate of the current mean and covariance.
- Kernel methods: Kernel methods owe their name to the use of kernel functions, which enable them to operate in a high-dimensional, implicit feature space without ever computing the coordinates of the data in that space, but rather by simply computing the inner products between the images of all pairs of data in the feature space.
- Additive manufacturing: The term "additive manufacturing" covers a variety of processes in which material is joined or solidified under computer control to create a three-dimensional object, with material being added together (such as liquid molecules or powder grains being fused together), typically layer by layer.

- Thermocouple: A thermocouple is an electrical device consisting of two dissimilar electrical conductors forming electrical junctions at differing temperatures. A thermocouple produces a temperature-dependent voltage as a result of the thermoelectric effect, and this voltage can be interpreted to measure temperature.
- Accelerometer: An accelerometer is a device that measures proper acceleration. Proper acceleration, being the acceleration (or rate of change of velocity) of a body in its own instantaneous rest frame, is not the same as coordinate acceleration, being the acceleration in a fixed coordinate system.
- Infrared sensor: An infrared sensor is an electronic instrument that is used to sense certain characteristics of its surroundings. It does this by either emitting or detecting infrared radiation. Infrared sensors are also capable of measuring the heat being emitted by an object and detecting motion.
- PHM: Prognostics and health management (PHM) is a framework that offers comprehensive yet individualized solutions for managing system health.
- ET: Extruder temperature measured by a thermocouple.
- BT: Building table temperature measured by a thermocouple.
- WT: Working temperature measured by a IR temperature sensor.
- MT: Meltpool temperature measured by a IR temperature sensor.
- BVX: Building table vibration at x-axis measured by an accelerometer.
- BVY: Building table vibration at y-axis measured by an accelerometer.
- BVZ: Building table vibration at z-axis measured by an accelerometer.
- EVX: Extruder arm vibration at x-axis measured by an accelerometer.
- EVY: Extruder arm vibration at y-axis measured by an accelerometer.
- EVZ: Extruder arm vibration at z-axis measured by an accelerometer.